The Dissertation Committee for Brian Christopher Gunter
certifies that this is the approved version of the following dissertation:

COMPUTATIONAL METHODS AND PROCESSING STRATEGIES FOR ESTIMATING EARTH'S GRAVITY FIELD

Committee:

Byron Tapley, Supervisor

John Ries

Robert van de Geijn

Wallace Fowler

Glenn Lightsey

COMPUTATIONAL METHODS AND PROCESSING STRATEGIES FOR ESTIMATING EARTH'S GRAVITY FIELD

by

BRIAN CHRISTOPHER GUNTER, B.S.E., M.S.E.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

December 2004

To Angela and Meghan

Acknowledgements

I would first like to acknowledge my committee supervisor, Byron Tapley, for providing me with the opportunity to be involved with the GRACE mission. The contributions of the other committee members were also greatly appreciated and helped to strengthen the quality of this document. I would like to thank John Ries, Srinivas Bettadpur, Steve Poole and Richard Eanes for the guidance and assistance they have provided over the past seven years. My appreciation to Don Chambers and Minkang Cheng for fielding countless requests to run their evaluation tests on the many solutions created as part of this work. Thanks also to Paul Thompson for helping with the variability plots and for providing some very useful simulation scripts. I want especially to thank Robert van de Geijn for serving as a friend and mentor these many years.

Support for this work was provided by the NASA Earth System Science Fellowship program and the Dolores Liebmann Fellowship program. Additional support for this research was provided by the University of Texas Center for Space Research. The author would also like to acknowledge the Texas Advanced Computing Center (TACC) for providing access and support to the various computational resources used in the development of this study.

Lastly, I would like to recognize my family, whose encouragement over the years has guided me through the many phases of this work. It is without exaggeration to say that I would not have gotten this far without them.

COMPUTATIONAL METHODS AND PROCESSING STRATEGIES FOR ESTIMATING EARTH'S GRAVITY FIELD

Publication No. _____

Brian Christopher Gunter, Ph.D.
The University of Texas at Austin, 2004

Supervisor: Byron D. Tapley

The focus of this study was to observe and characterize the behavior of certain types of errors present in the gravity field model estimation process, as they relate to fields created from the recently launched Gravity Recovery and Climate Experiment (GRACE). The instruments and configuration of the GRACE satellites are different from any other previously flown gravity mission, so the impact that these error sources have on the GRACE gravity solutions is not fully understood. The high resolution gravity perturbations detectable by GRACE also mean that many of these errors can only be fully explored through the use of high spherical harmonic degree and order solutions. When this study first began, a software estimation tool did not exist that was capable of handling the extremely large problem sizes that the GRACE mission can create. To address this issue, a parallel application called the Advanced Equation Solver for Parallel Systems (AESoP) was developed that was designed to accommodate the computational requirements of GRACE. An outline of the functionality and methodologies employed by AESoP is provided, as well as

detailed descriptions of the parallel algorithms created as part of its development. Using this new software tool, several types of errors inherent to the GRACE gravity field estimation process were analyzed. Investigations into the errors of omission and commission were performed using both real and simulated GRACE data. Additional studies into the combination of the GPS and inter-satellite ranging measurements were also conducted in an attempt to maximize the contribution of each data type as well as to improve processing efficiency. The results of these studies outline several processing strategies by which many of the error sources investigated can be significantly reduced while simultaneously decreasing the processing time and disk storage requirements by roughly 75% for an average GRACE solution.

Contents

List of Tables

List of Figures

Abbreviations

AESoP	Advanced Equation Solver for Parallel Systems
BLAS	Basic Linear Algebra Subprograms
CHAMP	CHAllenging Mini-satellite Payload
DV	Degree Variance
DDV	Degree Difference Variance
DEV	Degree Error Variance
FMR	Force Model Resolution
GPS	Global Positioning System
GPSDD	GPS Double-differenced
GRACE	Gravity Recovery and Climate Experiment
KBR	K-band Range
LLISS	Large Linear System Solver
MFLOPS	Millions of Floating-point Operations Per Second
MPI	Message Passing Interface
MSODP	Multi-Satellite Orbit Determination Program
NASA	National Aeronautics and Space Administration
OOC	Out-of-Core
PLAPACK	Parallel Linear Algebra Package
PR	Partials Range
PSD	Power Spectral Density
REF	Reference, or nominal, field
RMS	Root Mean Square
SGG	Satellite Gravity Gradiometry
SST	Satellite-to-Satellite Tracking

Chapter 1

Introduction

1.1 Problem Definition

The successful launch of the Gravity Recovery and Climate Experiment (GRACE) in March of 2002 brought the scientific community a new and extremely accurate source of data with which to model the Earth's gravity field. While this data has already been used to generate gravity field models that are 10 to 100 times more accurate than any previous model (for the long and medium wavelengths)[60], the computational requirements necessary to achieve these results are substantial. An annual gravity field model involves the least squares estimate of tens of thousands of parameters from terabytes worth of data. In addition, the unique characteristics of the GRACE instruments required that many of the standard error sources and processing techniques be scrutinized. Both of these issues present a number of challenges that this dissertation will address.

The determination of the Earth's gravity field is a complicated procedure and requires input from many different sources. Data must first be collected, then processed with the appropriate mathematical models, and finally tested to verify the results. Each stage of the process is full of imperfections and limitations. The quality of the data depends on the performance of the spacecraft (i.e., attitude and orbit control, mass trim mechanisms, etc.), as well the

accuracy and precision of the on-board instruments (accelerometers, antennas, etc.). The processes which govern the Earth's mass variation are not fully understood, so the mathematical models we use to represent them have many assumptions and simplifications built into them. The software used to compute the models, as well as the machines they are run on, have finite capabilities and precision. Each of these imperfections introduces a certain degree of error into the final gravitational model. The reduction of these errors happens incrementally, building on the knowledge gained from the past. The goal of this dissertation will be to reduce the effect of some of these error sources, thereby improving the quality of the gravity field models created with the GRACE data.

Before the analysis phase of this study could begin, it was necessary to expand the processing power and computational efficiency of the software used to generate the gravity models. Accumulating and solving for the gravity model coefficients is by far the most computationally intensive component of the GRACE data processing scheme. The memory, disk storage and overall compute cycles needed to create a typical GRACE monthly or annual solution (see Appendix B.1) far exceeds the capacity of conventional single processor or vector machines. The legacy software that was previously used [71] to perform the least squares accumulation task was a serial application designed to run on these smaller systems. As a result, the hardware capabilities of these machines, both in terms of memory and processing power, limited the size of problem that could be handled. Fortunately, the operations involved with the accumulation phase are well suited for use on parallel architectures, in which the combined resources of tens or hundreds of processors can be used to tackle

large problems. Consequently, the first objective of this work was to create a new parallel least squares estimation tool that could accommodate the large problem sizes and data volume created from GRACE. The development of this new parallel application began long before the GRACE mission launched and has experienced a number of enhancements and discoveries over the years. One of the more notable developments involved the creation of a new class of parallel algorithms designed to permit the solution of problems of arbitrary size. A complete description of these and other algorithms will be provided later, along with a summary of the capabilities available with the new parallel solver.

The GRACE mission's twin satellites contain many unique instruments that permit the recovery of high resolution gravity field models. The High Accuracy Inter-satellite Ranging System (HAIRS), which measures the relative distance between the satellites to an accuracy of 10 micrometers, is one example. The SuperSTAR accelerometer, designed to measure the non-gravitational forces acting on the satellites, is the also the most precise of its kind ever flown. Before the mission launched, there was concern that the precision of these instruments, as well as the new inter-satellite ranging observable, may bring to light many different error sources that were previously too small to be of importance. As a result, the influence of various processing choices, particularly those surrounding the batch estimation procedure, were investigated in an effort to understand their impact on the GRACE gravity field models. Examples of this include the influence of errors of omission and commission, which involve the assumptions and limitations of the nominal field used in the batch estimation process. The nature of these errors makes it difficult to distinguish them from the true gravity signal. For this reason, the exploration of these error sources

could only be performed through the use of simulations. The results of these simulations, to be presented later, show how each of the specific error sources investigated influence the gravity field solution. The simulations also show how all of the errors examined can be sufficiently mitigated through the appropriate choice of processing parameters.

Another related topic investigated in this work involved the way in which certain GRACE data components are processed. The combination of the GPS data, collected from each of the GRACE satellite's on-board Blackjack receivers, with the inter-satellite ranging data is a complex interaction that must be handled carefully in order to optimize the contribution of each data type. While the GPS data is important for the determination of the longest wavelength gravity signals and for satellite positioning, it is not sensitive to gravity perturbations beyond the low to mid degrees. The majority of the gravity signal is recovered through the much more sensitive inter-satellite K-band range (KBR) observable. To better understand the relationship between the GPS and KBR data, a number of experiments were conducted with the objective of finding the optimal combination method for these two data types. One important aspect of these experiments involved the size of the data files for each respective type. When processing of the first GRACE mission data sets began, the number of GPS double-differenced measurements involved in the solution process far exceeded the number of measurements created from the KBR data. At one point, over 90% of the disk space and compute cycles were being devoted to the storage and processing of the comparably less sensitive GPS data. Using both real and simulated GRACE data, a series of processing strategies will be described in which the GPS and KBR data can be combined in such a

manner that the quality of the gravity field model is actually improved, while also significantly reducing the size of the GPS data files involved.

1.2 Background

One technique for measuring the Earth's gravity field is to observe the path of an orbiting satellite. The variations in the mass and density of the Earth will perturb the orbit of a satellite, so by accurately measuring the position of the satellite as it flys over all of the Earth's surface, we can infer what gravitational forces might be acting on the satellite. By observing the orbits of many different satellites, with different inclinations, over a long period of time, geodesists over the past three decades have been able to create a reasonably precise gravity field model at the longer wavelengths . The current models still have many limitations built into them. The most important of these is the fact that only very nearly polar orbits will allow the solution of a gravity field from a single satellite. Nearly all of the satellites used for these earlier models had orbits with an inclination that was less than 70 degrees, leaving the poles untreated and not providing the global coverage needed to create a truly robust gravity model. The use of additional satellites or other constraints help reduce these observability problems, but it is still a fundamental limitation of the non-polar satellite configuration. Another limitation is the fact that the technique of using ground-based satellite ranging measurements to determine the satellite orbits can (at its current level of accuracy) only support a field of roughly 500 km spatial resolution, or 1 m geoid [51]. This includes the incorporation of surface and altimetry data, which have their own set of limitations. For many Earth science applications, this resolution is simply too large to be of value.

It was for these reasons and others that scientists since the late 1960's have proposed launching dedicated gravity field recovery missions that make use of the satellite-to-satellite tracking (SST) or satellite gravity gradiometry (SGG) concepts [53]. The basic notion behind the SST approach is to accurately measure the range between two orbiting satellites (high-low or low-low). By doing so, many of the observability problems of the single satellite configuration are resolved, and a much higher resolution gravity field model can be obtained. The SGG method relies on a single satellite recording small variations in the spacecraft's gravitational acceleration in space using an instrument known as a gradiometer.

Several mission proposals based on these principles were offered in the 1980's and 1990's, such as the USA's Geopotential Research Mission (GRM) [36] and the European GPS/ARISTOTELES [20] and Satellite Test of the Equivalence Principle (STEP) missions [54]; however, none were approved for launch. Building on the foundation of these earlier mission proposals, several new missions were finally accepted to launch at the start of the new millennium. They include the CHAllenging Mini-satellite Payload (CHAMP) [43], the Gravity Recovery and Climate Experiment (GRACE), and the Gravity Field and Steady-State Ocean Circulation Explorer (GOCE) [21] missions. Each are designed to observe different regions of the gravity signal spectrum using variations of the SST and SGG concepts.

In July of 2000, the CHAMP mission was first of these mission to launch and employed the high-low SST recovery method. As a single satellite, CHAMP utilizes the SST concept by computing accurate range and range-rate measurements with the high altitude Global Positions System (GPS) satellite constel-

lation (via an on-board GPS receiver). As a high-low SST mission, CHAMP is primarily involved with resolving the long and medium wavelength gravity signals. The GOCE mission, scheduled to launch in 2006, is based on the SGG method and will make use of a high-accuracy gradiometer to recover signal at the high end of the gravity spectrum.

The GRACE mission, a joint venture between the University of Texas at Austin, NASA, the GeoForschumZentrum Potsdam (GFZ), and the Deutches Zentrum für Luf and Raumfahrt (DLR), is based on the low-low SST method. Launched in March of 2002, the mission consists of two twin satellites flying in a coplanar orbit, separated by a distance of roughly 200 km. The absolute position of each satellite is monitored with on-board Blackjack GPS receivers (similar to those flown on CHAMP), and the satellites track their relative position through the use of a special K-band microwave tracking system, called the High Accuracy Inter-satellite Ranging System (HAIRS). The satellites are not in a free-fall environment, so the non-gravitational forces acting on the satellites are measured through the use of an ONERA SuperSTAR Accelerometer. By precisely measuring their position relative to each other, small variations in the Earth's gravity field can be detected. For example, as the two satellites approach a mountain range, the leading satellite will experience a slight acceleration first (due to the mountain's increased mass and gravitational attraction) and then the second satellite will realize the same acceleration moments later. Using this twin satellite configuration, both large and small gravity perturbations can be detected and distinguished from each other. In addition to the K-band ranging, the GRACE mission flies in a near-polar orbit (i.e., 89 degree inclination), providing valuable coverage of the polar regions as well as uniform

coverage of the rest of the globe. The low altitude of the satellites, starting at 500km and decaying to roughly 300km at the end of the mission lifetime, allows the satellites to detect gravity perturbations of a resolution up to 250 km. No other mission flown to date has been able to offer this level of coverage and detailed spatial and temporal gravity information. The early results achieved by the GRACE and CHAMP missions have already improved our understanding of the Earth's gravity field by orders of magnitude, and it is hoped that the contribution of the GOCE mission will allow further improvements to be realized.

1.3 Previous Studies

Much of the work presented here is the continuation of a collection of previous studies. The Multi-Satellite Orbit Determination Program (MSODP) [46] used to generate the measurement partials, both real and simulated, used for each case study has itself been updated over the years by the efforts of various individuals. Work by Bettadpur [3], Davis [11], Sharma [52], Kim [37] and others were all instrumental in developing the force and noise models currently implemented in MSODP, enabling the accurate simulation of the GRACE environment. Most of the analysis performed here would not have been possible without these contributions.

The development of the parallel processing tools used in this study followed directly from an earlier serial application created by Yuan [71]. Yuan's Large Linear System Solver (LLISS) served as the prototype for the updated software package presented here, and was used extensively for testing and verification purposes. The optimal weighting algorithms developed by Tapley et

al. [61, 64], and later expanded by Yuan [72], were also an integral part of the solution and evaluation process of this study.

The error studies were also a continuation of work initially begun by Sanso [47] and Jekeli [34]. Even though their studies focused on slightly different aspects of the broad topic of aliasing error in geodesy, their research was a valuable research aid and formed the foundation for the related work shown here.

1.4 Objectives

The primary objective of this work was to investigate the influence that various error sources have on the gravity field models created from GRACE mission data. Part of this included the investigation of different types of processing errors, such as the errors of omission and commission, through the use of simulation experiments. Related to these efforts was an analysis into the combination of the GPS and KBR measurements. The time and resources initially required to process the GPS data was disproportionately large, motivating the development of a new processing strategy by which the size of the GPS data files could be reduced while not affecting the quality of the resulting gravity field models. Both real and simulated GRACE data were used to demonstrate that this goal can be achieved by employing certain processing techniques.

Another objective of this study was to develop the software tools needed to fully explore the capabilities of the GRACE mission data. The precision and accuracy of the GRACE instruments permit the solution of high degree and order gravity field models; however, the computational resources needed to create these solutions on a regular basis can be substantial. To address this

issue, a considerable amount of effort was placed towards the development of a least squares solver designed to operate in a parallel environment. This new parallel software application was used to compute all of the gravity field solutions described in this study and now serves as one of the GRACE mission's core processing tools.

1.5 Outline

As was mentioned earlier, the work of this study was done in two stages. The organization of this dissertation will follow the same format. The first part will outline the algorithms and software used to generate a high degree and order gravity field model. The second part will detail how this software was used to explore the various processing errors described in Section 1.1.

Chapter 2 is devoted to the description of the software and algorithms used to conduct the experiments discussed in the later chapters. A high level description of the application design and methodology is then followed by a detailed description of the in-core and out-of-core least squares algorithms. Chapter 3 is dedicated to studying the effects of omission and commission errors on the gravity solutions. A number of simulations are described that detail the behavior of these errors, as well as methods available to reduce their influence on the gravity models. Chapter 4 is concerned with the combination of the GPS data and K-band range-rate (KBR) data. Both simulated and real data solutions are used to demonstrate how a substantial increase in processing efficiency can be achieved through the appropriate treatment of the GPS data. Finally, Chapter 5 provides a brief summary of each chapter's contribution and any conclusions reached. Recommendations for future research efforts are also

proposed.

1.6 Notation

In this dissertation, the following conventions have been adopted: Matrices, vectors, and scalars are denoted by upper-case, lower-case, and lower-case Greek letters, respectively. The identity matrix will be denoted by I and e_1 will denote the first column of the identity matrix (in other words, the vector with first element equal to unity and all other elements equal to zero). The dimensions and lengths of such matrices and vectors will generally be obvious from context.

Many of the algorithms in this paper are given in a notation that has been recently adopted as part of the Formal Linear Algebra Methods (FLAME) project [24, 42]. The double lines in the partitioned matrices and vectors relate to how far into the matrices and vectors the computation has proceeded, indicating which parts are in their factored or original form. It is hoped that the notation is intuitive, but suggest that the reader consult some of these related papers for further clarification.

Lastly, when referring to the spherical harmonic degree and order range of a gravity solution, an abbreviated notation will be used. For example, a degree and order 120 solution will be shortened to the expression 120x120.

Chapter 2

A Parallel Least Squares Solver

2.1 Introduction

The ability to process the large amount of data generated by GRACE (see Appendix B.1) and other missions requires the development of the proper software in addition to access to capable hardware. Modern advances in high performance computing have resulted in machines that combine high speed processors, large memories and disk storage, and high-bandwidth networking. However, the hardware is only valuable if the software is in place to take advantage of its power.

A significant amount of the computation surrounding the generation of a gravity field model involves the linear least squares estimation of tens of thousands of parameters using millions of observations. The least squares reduction is by nature an $O(n^3)$ operation and is rich in matrix-matrix operations. These types of operations are well suited for use in parallel architectures, and a substantial performance boost can be gained by operating in a parallel environment.

This was the primary motivation for the development of what has now matured into the Advanced Equation Solver for Parallel Systems (AESoP). Early development of the parallel algorithms [25, 31, 29] contained within AESoP began over five years ago, and since then AESoP has evolved into a

robust and efficient parallel application that can process extremely large data sets using complex parameterizations.

This chapter will outline the development and implementation of AESoP, in particular as it is applied to the solution of the GRACE gravity field models. A high level description of the history and development philosophy of the application will be given in Section 2.2. Sections 2.4-2.5 will provide detailed descriptions of the primary algorithms employed by AESoP. Sections 2.5.5 and 2.6 will provide implementation details along with performance statistics for the in-core and out-of-core algorithms. Finally, Section 2.7 will offer a summary and a few final remarks.

2.2 Development

It was mentioned in Chapter 1 that the need for the development of a new software tool, complete with new processing algorithms, was motivated primarily by the large number of information equations generated by the GRACE mission. The least squares estimation phase is by far the most computationally intensive part of the estimation process, so attention was focused on this particular component.

When work first began on this study, the existing linear solver program, called the Large Linear System Solver (LLISS) [72, 71], was a serial application designed to run on smaller single processor machines or parallel vector processor (PVP) machines, such as the Cray SV1. While the functionality of LLISS was more than sufficient to generate a gravity field model, both the application and the hardware it was designed to run on were not capable of handling the larger data sets and problem sizes associated with the GRACE mission. Therefore,

the original goal of AESoP was to take the functionality of LLISS and port this to a parallel environment. The following list summarizes the core functionality requirements under which AESoP was developed:

- Compute a linear least squares solution using orthogonal transformations.
- Store the resulting accumulations in a reusable file format.
- On each iteration, calculate an optimal weight based on post-fit residuals for each input data file.
- Allow for parameters to be solved within varying timespans.
- Permit the use of *a priori* conditioning on the system.
- Shift an equation set to a given reference field when the set has been created with different nominal fields.
- Compute the error covariance of the resulting solution.

In addition to these basic requirements, the implementation of AESoP had to have good performance and the ability to run on different platforms. The state-of-the-art in high performance computing is constantly improving, and every few years new machines become available that provide substantially more power than their predecessors. These new machines may not always be created by the same vendor or use the same operating system, which is why portability is so important.

The need for high performance implementations is obvious, as even the fastest machines will run slowly with poorly designed software. In some in-

stances, machine specific enhancements or code libraries can be used to improve the performance of the algorithms presented later, but the price for this is a greater reliance on a particular machine and vendor. It also tends to make the code implementation more complex. The approach that was taken in the development of AESoP was to balance performance with the highest degree of portability. Fortunately, there exist many standard parallel computing libraries that are available on most high performance systems. These include packages such as the Message Passing Interface (MPI) [23, 55] and the Basic Linear Algebra Subprograms (BLAS) [39, 13] libraries. These core libraries provide the foundation for most parallel applications, as they handle the basic processor communication and linear algebra operations, and are often highly optimized for the architecture on which they are installed.

In addition to these, AESoP made extensive use of the Parallel Linear Algebra Package (PLAPACK)[67, 8, 1], an infrastructure for building highly optimized linear algebra libraries. PLAPACK's unique 'view-based' infrastructure handles most of the intricacies of matrix indexing and processor communication, letting the developer focus more on the algorithms and less on the details of the implementation. PLAPACK itself relies on the use of standard BLAS and MPI libraries, providing the speed and portability required by AESoP.

The C programming language was chosen as the implementation tool for AESoP because of its widespread availability and functionality. The ability to easily construct objects, manage complex memory operations, and conduct efficient I/O transfers are just a few items for which the C language is particularly well suited and would be used regularly within AESoP.

Figure 2.1 illustrates the relationship between AESoP and its dependen-

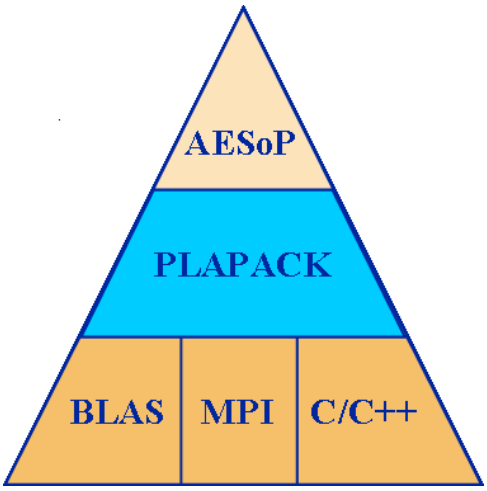


Figure 2.1: An illustration of the AESoP Software Hierarchy

cies. These dependencies are the foundation for AESoP and provide a majority of its functionality. In addition to these core libraries, there are other requirements that must be met in order for AESoP to run properly. Details such as file formats and user input options all must be defined properly. Unlike the core libraries, these are all flexible and can be modified to suit a given purpose. The large number of user options and formatting requirements of AESoP, as well as the fact that the software is constantly undergoing enhancements, makes it impractical to list them all in this document. Instead, the reader should refer to the AESoP User's Manual [26].

2.3 Application Design

While the previous section outlined the functional requirements for AESoP, considerable leeway was available on how these could be implemented. The primary objective of the application design was to make the code flexible. To do this, a number of design choices were made to facilitate flexibility and encourage

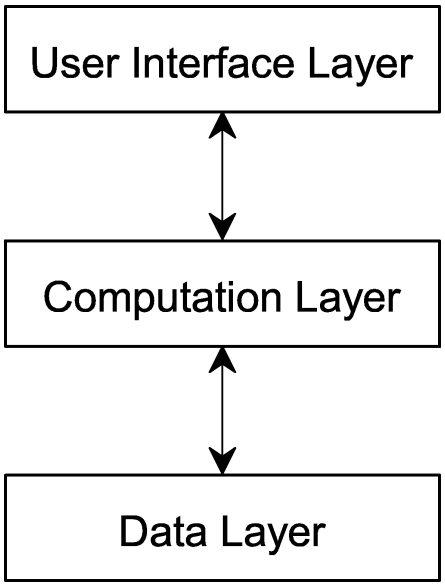


Figure 2.2: The three-tiered architecture employed by AESoP.

code re-use.

The most important of these was to implement a three-tiered software architecture. This is a common methodology used in developing frameworks and other software applications, and is designed to intentionally separate various components of the application in order to reduce the impact of code changes. Figure 2.2 depicts the three different layers used in the development of AESoP. The three primary layers are the user interface layer, the computation layer, and the data layer.

The user interface layer handles all inputs from the user regarding how the software is to be used. This includes a listing of which files to process, the parameterization of the solution, plus a number of other settings describing the action should be taken for the given run (i.e., use *a priori* information, compute the covariance, etc.; see [26] for a complete list of available options). The code

in this layer is responsible for collecting the details of the job to be run and passing them to the computation layer (in the form of predefined structures, or objects). The computation layer is the part of the application that, as its name implies, performs the computations and operations requested by the user. For AESoP, this is the layer that performs the least squares calculations, shifting, and optimal weighting techniques mentioned earlier. Finally the data layer handles all transactions between the computation layer and the data files. It should be noted that the lines of communication between the layers are limited. As Figure 2.2 implies, the user interface layer does not interact directly with the data layer. This is by design and, as the next discussion explains, has a useful purpose.

Having each component in a separate layer is beneficial for a number of reasons. The main reason is to reduce the impact of changes to the code. If done properly, changes to one layer should have no impact on the other layers. For example, the user interface for AESoP is currently limited to an ASCII input file, but this could be converted to a Graphical User Interface (GUI) at some point in the future. This change would only impact the routines in the user interface layer, because regardless of how the user input is collected, the same information will be passed to the computation layer. Similarly, the data layer is its own component because file formats often change, either because the user changes them purposely or because the code is being run on different platforms with different binary formats. By having limited channels to the data files, changes such as these can be easily accommodated without affecting the user interface or the software algorithms (i.e., the computation layer). This feature, in particular, has helped make porting AESoP from one platform to

another considerably easier.

2.3.1 Object-based approach

Another important design choice made in the development of AESoP was to adopt an object-based coding approach. Related information was grouped and arranged into pre-defined structures, or objects. These objects can themselves be grouped and combined to form other objects. Objects can be passed from one routine to the next with the philosophy that related information will often be easily accessible. This differs from true object-oriented programming, which makes use of many more advanced object features, such as inheritance and polymorphism. Nonetheless, the code arrangement is such that the transition to a true object-oriented environment could be considered in future versions of AESoP.

The primary benefit of using the object-based approach is that it encourages code re-use. By having the various modules operate as independent units with objects as argument list inputs, the code becomes more generic in scope and can be used in a variety of different applications. So while AESoP was designed specifically to create gravity field models from GRACE, the code could easily be modified to work on any type of least squares application. This flexibility also makes enhancements and other changes easier to implement.

2.3.2 Parameter Leveling

Often when experimenting with different parameterizations for a given data set, the need arises to vary the scope or context over which a given parameter or set of parameters is solved. For example, a typical GRACE data set for

a given day consists of both GPS and KBR data (i.e., measurement partials and observation), each stored in separate files. Each file contains parameters that are common to both files and some that are particular to the individual data file. In addition, some parameters may only be valid over a single arc (which in this example is defined to be one day), while others will be estimated over multiple arcs. For this reason, the parameters in a given data file are categorized into three different groups, or levels:

- **Local** : Parameters that are valid for only one arc and one file.
- **Common** : Parameters that are valid across multiple arcs and files.
- **Global** : Parameters that are valid across all arcs and all files.

The mechanism to deal with these different parameter groups is called *parameter leveling*. There may be multiple layers of the local and common levels, depending on the definition of the arc length. For example, a month's worth of data may have common parameters defined at the day, week, and month time spans. AESoP gives the user full control over the scope and context of every parameter to be estimated [26].

The terms local, common and global parameters will be used throughout the text and will refer to these three parameter leveling groups. Typical examples of local parameters would include parameters such as the KBR low-low biases, GPS ambiguity parameters and GPS zenith delay parameters. Common parameters are typically items such as the satellite's initial conditions and accelerometer biases, while the global parameters are usually the spherical harmonic coefficients and accelerometer scales. Again, these are merely examples

and, depending on the goal of the experiment, each of the parameter types just mentioned could have been placed in any of the parameter leveling groups.

2.4 In-core Algorithms

The goal of this section is to discuss the algorithms that are implemented in AESoP, in particular those that pertain directly to the creation of a gravity field model from a given set of linear equations. The most important of these is the least squares reduction algorithm. This algorithm is responsible for the vast majority of calculations in the estimation process, so it shall be covered in considerable detail, beginning with the fundamental concepts and progressing through to the more advanced blocked algorithms. The description of most of the algorithms of this section and the next have already appeared in a previous study or publication [25, 29, 30], and are repeated here for completeness.

Note 1 *Additional algorithms used by AESoP, such as optimal weighting and shifting, can be found in Appendix A.3.*

2.4.1 The Normal Equations

The primary function of least squares estimation is to fit a model to a set of observations that is known to have errors. The process is designed to estimate the values of the model parameters such that the error between the observed and computed observations is minimized. For example, given the following system

$$Ax = y; \quad A \in \Re^{m \times n}; \quad x, y \in \Re^{n \times 1}; \quad m \geq n$$

it is desired to find the parameters, x, that come closest (by some predefined metric) to representing the observed measurements in y. One way to accomplish this is to treat the system as an optimization problem and define a performance index that can then be minimized. For the least squares method, the performance index is chosen to be the sum of squares of the residuals, or errors. Defining the error to be

$$\epsilon = Ax - y$$

the performance index becomes

$$J(x) = \epsilon^T \epsilon \tag{2.1}$$

For the case of weighted least squares, the weight matrix

$$W = \begin{bmatrix} w_i & 0 & \cdots & 0 \\ 0 & w_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_n \end{bmatrix}$$

can be introduced into Eqn. 2.1 to obtain

$$\begin{aligned} J(x) &= \epsilon^T W \epsilon && (2.2) \\ &= [Ax - y]^T W [Ax - y] && (2.3) \\ &= [x^T A^T W - y^T W][Ax - y] \\ &= x^T A^T W A x - x^T A^T W y - y^T W A x + y^T W y \end{aligned}$$

Minimizing the performance index is done by taking the 1^{st} variation of the performance index (i.e., setting $\frac{\partial J(x)}{\partial x} = 0$).

$$\frac{\partial J(x)}{\partial x} = -2y^T W A + 2x^T A^T W A = 0 \tag{2.4}$$

This can be solved to get

$$\begin{aligned} 2x^T A^T W A &= 2y^T W A \\ (A^T W A)x &= A^T W y \end{aligned} \tag{2.5}$$

The result, Eqn. 2.5, is commonly referred to as the Normal Equations. If A consists of at least n linearly independent observations, then the *Normal Matrix*, $A^T W A$, is both symmetric and positive definite. That condition also implies that the inverse $(A^T W A)^{-1}$ exists, allowing a solution for x.

2.4.2 The QR Factorization

The method of Normal Equations is just one technique commonly used to solve least squares systems. An alternative to this approach, called the QR Factorization, involves the introduction of the orthogonal transformation

$$W^{1/2} A = QR,$$

where $Q \in \Re^{mxm}$ is orthogonal and R is upper triangular (see Tapley et al. [63] or Bjorck [5]). By definition, the matrix Q is said to be orthogonal if

$$Q^T Q = Q Q^T = I \tag{2.6}$$

By inserting Eqn. 2.6 into Eqn. 2.3,

$$\begin{aligned} J(x) &= [Ax - y]^T W^{1/2} Q Q^T W^{1/2} [Ax - y] \\ &= (Q^T W^{1/2} [Ax - y])^T (Q^T W^{1/2} [Ax - y]) \\ &= \|Q^T W^{1/2} [Ax - y]\|_2 \\ &= \|Q^T W^{1/2} Ax - Q^T W^{1/2} y\|_2 \end{aligned}$$

If Q is chosen such that

$$Q^T W^{1/2} A = \begin{bmatrix} R \\ 0 \end{bmatrix} \qquad Q^T W^{1/2} y = \begin{bmatrix} b \\ e \end{bmatrix}$$

where R is upper triangular, then the least squares performance index becomes

$$\begin{aligned} J(x) &= \left\| \begin{bmatrix} R \\ 0 \end{bmatrix} x - \begin{bmatrix} b \\ e \end{bmatrix} \right\|_2 \\ &= \|Rx - b\|_2 + \|e\|_2 \end{aligned}$$

If it is again assumed that there are at least n linearly independent observations in A, R is nonsingular and the system

$$Rx = b$$

will have a unique solution that will minimize the performance index. Note that $\|e\|_2$ is a constant independent of x, so it cannot play a part in the minimization of the solution. Since R is upper triangular, once the QR factorization has been calculated, x can be obtained through simple back substitution.

2.4.3 Computing the QR factorization via Householder Transformations

There are several different methods for computing the QR factorization, including those based on Givens rotations, orthogonalization via Gram-Schmidt and Modified Gram-Schmidt, and Householder transformations [33, 22, 69]. For dense matrices, the method of choice depends largely on how the factorization is subsequently used, the stability of the system, and the dimension of the matrix. For problems where $m \gg n$, the method based on Householder transformations is typically the most appropriate algorithm, especially when Q does not need to be explicitly computed.

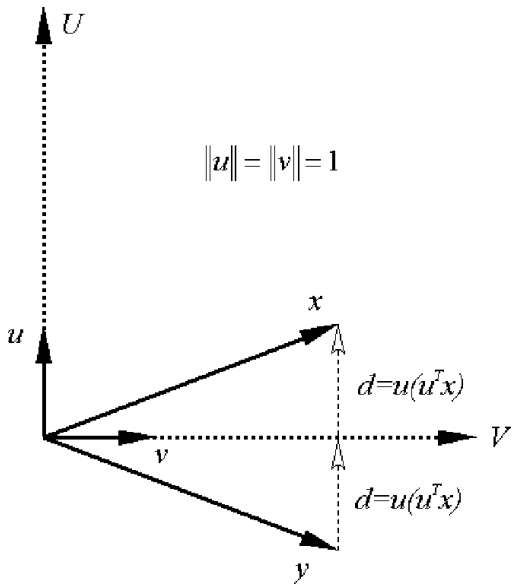


Figure 2.3: Householder Reflection

The simplest way to visualize the operation of the Householder reflection is to examine the two-dimensional vector case. Let u, v represent a unit basis for $\Re^2$, with $x \in \Re^2$. The basic idea is to reflect the vector x (which will later assume the role of a matrix column) about the axis V (See Fig. 2.3). Defining the vector d to be the projection of x onto u,

$$\begin{aligned} d &= \left(\frac{x \cdot u}{\|u\|_2^2}\right) u \\ &= (x^T u)u \\ &= u(u^T x) \end{aligned}$$

the following result is found through vector addition.

$$\begin{aligned} y &= x - 2d \\ &= x - 2u(u^T x) \\ &= (I - 2uu^T)x \end{aligned}$$

The matrix $Q = (I - 2uu^T)$ is called the *Householder transformation matrix*. It reflects x through the axis V. This same notion may be applied to the general n-dimensional, non-normalized case, stated as follows:

Generalized Householder Reflector:
Given $u \neq 0 \in R^n$ and defining $\beta = \frac{2}{\|u\|_2^2}$, then the reflector $Q = I - \beta uu^T$.

Note that Q is simply a rank one update of the identity matrix. It is easily verified that the matrix Q satisfies the condition of an orthogonal matrix.

$$\begin{aligned}
QQ^T &= (I - \beta uu^T)(I - \beta uu^T)^T \\
&= (I - \beta uu^T)(I - \beta uu^T) \qquad \text{Note: } (uu^T)^T = uu^T \\
&= I - 2\beta uu^T + \beta^2 uu^T uu^T \\
&= I - 2\left(\frac{2}{u^T u}\right) uu^T + \left(\frac{2}{u^T u}\right)^2 uu^T uu^T \\
&= I - \left(\frac{4uu^T}{u^T u}\right) + \left(\frac{4uu^T}{u^T u}\right) \\
&= I
\end{aligned}$$

In addition to orthogonality, Householder matrices are symmetric ($Q^T = Q$) and involutions ($Q^{-1} = Q$).

The ability to reflect any given vector through a chosen axis is useful in the QR factorization because it allows one to take a column vector and reflect it across a carefully chosen axis so that only the first element of the reflection is non-zero. For example, if it is desired to transform the vector

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad \text{into the vector} \quad y = \begin{bmatrix} \sigma \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

let $\sigma = \pm\|x\|_2$ so that $\|x\|_2 = \|y\|_2$. Since $x \neq y$, the definition above implies that there exists a transformation matrix Q such that $Qx = y$. A non-normalized basis vector, u, can be found through the vector subtraction of x and y.

$$u = x - y = [\ (x_1 - \sigma) \quad x_2 \quad \cdots \quad x_n]^T$$

Letting $\beta = \frac{2}{\|u\|_2^2}$, the transformation matrix $Q = (I - \beta uu^T)$ is easily calculated.

The process of triangularizing an entire matrix requires the application of a series of Householder transformations to zero out each column below the diagonal. In this respect, the triangularization of a generic $m \times n$ matrix, A, can be visualized as a product of transformations

$$Q^T A = Q_n^T \cdots Q_3^T Q_2^T Q_1^T A = R$$

where

$$Q_i = \begin{bmatrix} I & \\ & \hat{Q} \end{bmatrix} \qquad 1 \leq i \leq n$$

is the Householder transformation which zeros out the elements below the diagonal in the i^{th} column. This ability to quickly and easily zero the elements below the diagonal provides a powerful tool for triangularizing a matrix, and will serve as the driving engine for the QR transformation.

Given the real-valued vector x of length m, partitioned as follows

$$x = \left(\frac{\chi_1}{x_2}\right)$$

where χ_1 equals the first element of x, the basic algorithm for computing the Householder vector is described in Algorithm 2.4.1.

Algorithm 2.4.1 *Householder Vector*

Let $m = \|x\|_2$
if $m = 0$
  then $\beta = 0$
  else
    $\eta = -\text{sign}(\chi_1)\|x\|_2$
    $\nu_1 = \chi_1 - \eta$
    $u = \begin{pmatrix} 1 \\ (x_2/\nu_1) \end{pmatrix}$
    $\beta = \frac{2}{u^T u}$
fi

Computed this way, the transformation $(I - \beta uu^T)x = \eta e_1$ annihilates all but the first element of x. Note that Q is not explicitly created in Algorithm 2.4.1, requiring only β and u to be stored in the event the same transformation would like to be applied to another vector. In the following sections, the notation $[u, \eta, \beta] := h(x)$ will be used to represent the computation of the above mentioned η, u, and β from vector x, and the notation $H(x)$ for the transformation $(I - \beta uu^T)$ where $[u, \eta, \beta] = h(x)$.

The above procedure takes $O(n)$ floating point operations (flops) to compute. To apply a single transformation to an $m \times n$ matrix, A, primarily involves BLAS level-2 operations (i.e., matrix-vector multiplication, rank-one updates, etc.).

$$Q^T A = (I - \beta uu^T)A = A - \beta uu^T A = A - \beta u(A^T u)^T$$

Computed this way, a single Householder reflection update requires $4mn$ flops. The complete factorization $A = QR$ requires the application of n Householder matrices, and takes roughly $2n^2(m - n/3)$ flops ($4n^3/3$ flops for $m = n$).

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$ and $b \rightarrow \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right)$

where A_{TL} is 0×0 and b_T has 0 elements

while $n(A_{BR}) \neq 0$ **do**

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right) \text{ and } \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

where α_{11} and β_1 are scalars

$$\left[\left(\begin{array}{c} 1 \\ \hline u_2 \end{array}\right), \eta, \beta_1\right] := h\left(\begin{array}{c} \alpha_{11} \\ \hline a_{21} \end{array}\right)$$

$\alpha_{11} := \eta$

$a_{21} := u_2$

$w^T := a_{12}^T + u_2^T A_{22}$

$$\left(\begin{array}{c} a_{12}^T \\ \hline A_{22} \end{array}\right) := \left(\begin{array}{c} a_{12}^T - \beta_1 w^T \\ \hline A_{22} - \beta_1 u_2 w^T \end{array}\right)$$

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & a_{01} & A_{02} \\ \hline a_{10}^T & \alpha_{11} & a_{12}^T \\ \hline A_{20} & a_{21} & A_{22} \end{array}\right) \text{ and } \left(\begin{array}{c} b_T \\ \hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline \beta_1 \\ \hline b_2 \end{array}\right)$$

enddo

Figure 2.4: Unblocked Householder QR factorization.

2.4.4 A simple algorithm for the QR factorization via Householder transformations

The computation of the QR factorization commences as described in Figure 2.4. The idea is that Householder transformations are computed to successively annihilate elements below the diagonal of matrix A one column at a time. The Householder vectors are stored below the diagonal over the elements of A that have been so annihilated. Upon completion, matrix R has overwritten the upper triangular part of the matrix, while the Householder vectors are stored in the lower trapezoidal part of the matrix. The scalars β discussed above are

stored in the vector b.

If the matrix Q is explicitly desired, it can be formed by computing the first n columns of $H_1 H_2 \cdots H_n$ where H_i equals the i^{th} Householder transformation computed as part of the factorization described above. For the applications of this study, however, there is no need to form Q explicitly and thus the issue will not be treated further.

2.4.5 Block Algorithms

Until now, the concern has only been with the basic theory behind the techniques of least squares estimation, and not performance. The algorithms outlined up to this point have consisted primarily of BLAS level-2 operations. By their nature, BLAS level-2 operations are slower and less efficient to compute than the matrix-matrix operations of BLAS level-3. Therefore, it would be advantageous to develop techniques that utilize the more optimized matrix-matrix operations as much as possible. This implies a block approach.

2.4.6 A high-performance blocked algorithm for the QR factorization

It is well-known that high performance can be achieved in a portable fashion by casting algorithms in terms of matrix-matrix multiplication [2, 13, 15]. There exist many different block implementations for the QR transformation [5, p. 226]. Earlier implementations of the parallel blocked QR transformation done for this study [25] used the 'WY' transformation [4], but was later changed to use the compact variation of this algorithm developed by Schreiber and Van Loan [49]. The compact WY transformation offers improved performance and is also better suited to the out-of-core algorithm to be discussed later. The basic

premise of Schreiber and Van Loan's block Householder method is to compute a series of k Householder vectors and apply them to A using the transformation

$$Q = I + YTY^T \tag{2.7}$$

where T is a $k \times k$ upper triangular matrix and Y is a $n \times k$ lower (unit) trapezoidal matrix. Recall that, in the original algorithm, Q_i was shown to be a rank-one update of the identity matrix. The rank-k block representation of Q shown above is merely an extension of this. The following is an illustration of how Householder transformations can be combined into the matrix form of Equation 2.7. First, it will be assumed that the addition of a new Householder transformation, $P = I + \beta uu^T$, will take the form

$$\widetilde{T} = \begin{pmatrix} T & t \\ 0 & \tau \end{pmatrix} \qquad \widetilde{Y} = \begin{pmatrix} Y & u \end{pmatrix}$$

where t and τ represent the new column of T corresponding to P. The new blocked transformation, $\widetilde{Q}$, then becomes.

$$\begin{aligned} \widetilde{Q} &= I + \widetilde{Y}\widetilde{T}\widetilde{Y}^T \\ &= I + \begin{pmatrix} Y & u \end{pmatrix} \begin{pmatrix} T & t \\ 0 & \tau \end{pmatrix} \begin{pmatrix} Y^T \\ u^T \end{pmatrix} \\ &= I + YTY^T + Ytu^T + \tau uu^T \end{aligned} \tag{2.8}$$

Alternatively, the application of Q to P is as follows.

$$\begin{aligned} \widetilde{Q} &= QP \\ &= (I + YTY^T)(I - \beta uu^T) \\ &= I + YTY^T - \beta YTY^T uu^T - \beta uu^T \end{aligned} \tag{2.9}$$

Equations 2.8 and 2.9 are equivalent provided that $t = -\beta TY^T u$ and $\tau = -\beta$. This demonstrates how a series of rank-1 Householder transformations can be represented collectively as a single rank-k update to the identity. The algorithm used to generate the T and Y matrices, shown below in Algorithm 2.4.2, also follows directly from the above illustration.

Algorithm 2.4.2 *YTY^T Transform (Block Householder)*

for $j = 1 : r$
  if $j = 1$
    then
      $Y = [u_j]$
      $T = [-\beta_j]$
    else
      $t = -\beta_j TY^T u_j$
      $\tau = -\beta_j$
      $T = \begin{bmatrix} T & t \\ 0 & \tau \end{bmatrix}$
      $Y = [Y \;\; u_j]$
  fi
end

Note that Y is simply the collection of Householder vectors and is unit lower triangular. Because the creation of the T matrix is an added cost, this algorithm is slightly more expensive to compute than the original one, but this penalty is more than compensated for by the introduction of BLAS level-3 operations. Figure 2.5 describes how the block factorization can be applied to a generic matrix, A. The performance of this algorithm in a parallel environment will be discussed later in Section 2.6.

Note 2 *Notice that the algorithm stores the "T" matrices that are part of the block Householder transformation $I + YTY^T$. This avoids having to recompute*

Partition $A \rightarrow \left(\begin{array}{c||c} A_{TL} & A_{TR} \\ \hline\hline A_{BL} & A_{BR} \end{array}\right)$ and $T \rightarrow \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right)$

where A_{TL} is 0×0 and T_T has 0 rows

while $n(A_{BR}) \neq 0$ **do**

Determine block size k

Repartition

$$\left(\begin{array}{c||c} A_{TL} & A_{TR} \\ \hline\hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c||c|c} A_{00} & A_{01} & A_{02} \\ \hline\hline A_{10} & A_{11} & A_{12} \\ \hline\hline A_{20} & A_{21} & A_{22} \end{array}\right) \text{ and } \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right) \rightarrow \left(\begin{array}{c} T_0 \\ \hline\hline T_1 \\ \hline\hline T_2 \end{array}\right)$$

where A_{11} is $k \times k$ and T_1 has k rows

$$\left[\left(\begin{array}{c} A_{11} \\ \hline\hline A_{21} \end{array}\right), \eta, b_1\right] := \left[\left(\begin{array}{c} \{Y\backslash R\}_{11} \\ \hline\hline Y_{21} \end{array}\right), \eta, b_1\right] = QR\left(\left(\begin{array}{c} A_{11} \\ \hline\hline A_{21} \end{array}\right)\right)$$

Compute T_1 from $\left[\left(\begin{array}{c} \{Y\backslash R\}_{11} \\ \hline\hline Y_{21} \end{array}\right), \eta, b_1\right]$

$$\left(\begin{array}{c} A_{12} \\ \hline\hline A_{22} \end{array}\right) := \left(I + \left(\begin{array}{c} Y_{11} \\ \hline\hline Y_{21} \end{array}\right) T_1^T \left(\begin{array}{c|c} Y_{11}^T & Y_{21}^T \end{array}\right)\right) \left(\begin{array}{c} A_{12} \\ \hline\hline A_{22} \end{array}\right)$$

Continue with

$$\left(\begin{array}{c||c} A_{TL} & A_{TR} \\ \hline\hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c||c} A_{00} & A_{01} & A_{02} \\ \hline\hline A_{10} & A_{11} & A_{12} \\ \hline\hline A_{20} & A_{21} & A_{22} \end{array}\right) \text{ and } \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right) \leftarrow \left(\begin{array}{c} T_0 \\ \hline\hline T_1 \\ \hline\hline T_2 \end{array}\right)$$

enddo

Figure 2.5: Blocked Householder QR factorization.

those matrices as part of the out-of-core implementation of the QR factorization, and results in a small but noticeable increase in performance [18, 19]. While not implemented for this study, further optimizations can be gained for certain types of problems by formulating the T matrices in terms of Level-3 operations [19] as opposed to the traditional method which only incorporates Level-2 operations [49].

2.4.7 Solving multiple Linear Least-Squares problems

As mentioned earlier, given a real-valued $m \times n$ matrix A and vector y of length m, the linear least-squares problem is generally stated as

$$\min_x \|y - Ax\|_2$$

where the desired result is a vector x that minimizes the above expression. The minimizing vector, x, can be found by computing the QR factorization $A = QR$, computing $z = Q^T y$, and solving $Rx = z_T$ where z_T denotes the first n elements of z.

Alternatively, one can think of this as follows: Append y to A to form $\left(\; A \;\middle|\; y \; \right)$. Compute the QR factorization $A = QR$, storing the Householder vectors and R over A. Update y by applying the Householder transformations used to compute R to vector y, which overwrites y with z. Finally, solve $Rx = z_T$ with the first n elements of the updated y. This second approach is reminiscent of how a linear system can be solved by appending the right-hand-side vector to the system and performing an LU factorization (or, equivalently, Gaussian elimination) on the appended system, followed by a back-substitution step.

Partition $A \rightarrow \left(\begin{array}{c|\|c} A_{TL} & A_{TR} \\ \hline\hline A_{BL} & A_{BR} \end{array}\right)$, $B \rightarrow \left(\begin{array}{c} B_T \\ \hline\hline B_B \end{array}\right)$ and $T \rightarrow \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right)$

where A_{TL} is 0×0 and B_T and T_T have 0 rows

while $n(A_{BR}) \neq 0$ **do**

Determine block size k

Repartition

$$\left(\begin{array}{c|\|c} A_{TL} & A_{TR} \\ \hline\hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|\|c|c} A_{00} & A_{01} & A_{02} \\ \hline\hline A_{10} & A_{11} & A_{12} \\ \hline A_{20} & A_{21} & A_{22} \end{array}\right),$$

$$\left(\begin{array}{c} B_T \\ \hline\hline B_B \end{array}\right) \rightarrow \left(\begin{array}{c} B_0 \\ \hline\hline B_1 \\ \hline B_2 \end{array}\right), \text{ and } \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right) \rightarrow \left(\begin{array}{c} T_0 \\ \hline\hline T_1 \\ \hline T_2 \end{array}\right)$$

where A_{11} is $k \times k$ and B_1 and T_1 have k rows

$$\left(\begin{array}{c} B_1 \\ \hline B_2 \end{array}\right) := \left(I + \left(\begin{array}{c} Y_{11} \\ \hline Y_{21} \end{array}\right) T_1^T \left(\begin{array}{c|c} Y_{11}^T & Y_{21}^T \end{array}\right)\right) \left(\begin{array}{c} B_1 \\ \hline B_2 \end{array}\right)$$

NOTE: Here Y_{11} refers to the lower triangular part of A_{11} and Y_{21} to A_{21}

Continue with

$$\left(\begin{array}{c|\|c} A_{TL} & A_{TR} \\ \hline\hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|\|c} A_{00} & A_{01} & A_{02} \\ \hline A_{10} & A_{11} & A_{12} \\ \hline\hline A_{20} & A_{21} & A_{22} \end{array}\right),$$

$$\left(\begin{array}{c} B_T \\ \hline\hline B_B \end{array}\right) \leftarrow \left(\begin{array}{c} B_0 \\ \hline B_1 \\ \hline\hline B_2 \end{array}\right), \text{ and } \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right) \leftarrow \left(\begin{array}{c} T_0 \\ \hline T_1 \\ \hline\hline T_2 \end{array}\right)$$

enddo

Figure 2.6: Blocked forward substitution-like of right-hand-side matrix B.

Finally, if there exists a set of right-hand-sides, one can simultaneously solve a linear least-squares problem with A and columns of B by the following approach: Append B to A to form $\left(\begin{array}{c|c} A & B \end{array} \right)$. Compute the QR factorization $A = QR$, storing the Householder vectors and R over A. Update B by applying the Householder transformations used to compute R to matrix B, which overwrites B with Z. Finally, solve $RX = Z_T$ with the first n rows of the updated B. It is this second operation with a right-hand-side B that will be encountered in the out-of-core implementation of the QR factorization. An algorithm for the first, forward substitution-like, step is given in Figure 2.6.

Note 3 *Again, because the "T" matrices that are part of the block Householder transformation $I + YTY^T$ are stored in memory, they need not be recomputed as part of the "forward substitution" step on matrix B.*

2.4.8 Updating the QR Factorization

Frequently, the linear equations used in the least squares problem are collected incrementally. For example, if the observations from a particular instrument are only collected or contributed on a monthly basis, it would be useful to combine each new batch of data into the existing solution without having to recombine all of the previous data again. The following is a review of how the QR factorization can be updated as additional batches of equations (i.e., observations) become available [16, 22, 69].

2.4.9 Appended Data Factorization

Assume now that Q and R have been computed such that $A = QR$, overwriting A with the Householder vectors and upper triangular matrix R, and storing

Partition $R \rightarrow \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline\hline R_{BL} & R_{BR} \end{array}\right)$, $C \rightarrow \left(\begin{array}{c|c} C_L & C_R \end{array}\right)$, and $b \rightarrow \left(\begin{array}{c} b_T \\ \hline\hline b_B \end{array}\right)$

where R_{TL} and C_L are 0×0 and b_T has 0 elements

while $n(R_{BR}) \neq 0$ **do**

Repartition

$$\left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline\hline R_{BL} & R_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline\hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline\hline R_{20} & r_{21} & R_{22} \end{array}\right),$$

$$\left(\begin{array}{c|c} C_L & C_R \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} C_0 & c_1 & C_2 \end{array}\right), \text{ and } \left(\begin{array}{c} b_T \\ \hline\hline b_B \end{array}\right) \rightarrow \left(\begin{array}{c} b_0 \\ \hline\hline \beta_1 \\ \hline\hline b_2 \end{array}\right)$$

where ρ_{11} and β_1 are scalars and c_1 is a column

$$\left[\left(\begin{array}{c} 1 \\ \hline\hline u_2 \end{array}\right), \eta, \beta_1\right] := h\left(\begin{array}{c} \rho_{11} \\ \hline\hline c_1 \end{array}\right)$$

$\rho_{11} := \eta$

$c_1 := u_2$

$w^T := r_{12}^T + u_2^T C_2$

$$\left(\begin{array}{c} r_{12}^T \\ \hline\hline C_2 \end{array}\right) := \left(\begin{array}{c} r_{12}^T - \beta_1 w^T \\ \hline\hline C_2 - \beta_1 u_2 w^T \end{array}\right)$$

Continue with

$$\left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline\hline R_{BL} & R_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline\hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline\hline R_{20} & r_{21} & R_{22} \end{array}\right),$$

$$\left(\begin{array}{c|c} C_L & C_R \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} C_0 & c_1 & C_2 \end{array}\right), \text{ and } \left(\begin{array}{c} b_T \\ \hline\hline b_B \end{array}\right) \leftarrow \left(\begin{array}{c} b_0 \\ \hline\hline \beta_1 \\ \hline\hline b_2 \end{array}\right)$$

enddo

Figure 2.7: Unblocked update to a QR factorization.

the "T" matrices in matrix T. Thus, the quantities $A = \{Y \backslash R\}$ and T are available. Now, consider the QR factorization of matrix

$$\begin{pmatrix} A \\ C \end{pmatrix} = \bar{Q}\bar{R} \tag{2.10}$$

A key observation is that the QR factorization of

$$\begin{pmatrix} R \\ C \end{pmatrix} \tag{2.11}$$

produces the same upper triangular matrix $\bar{R}$ as does the factorization in (2.10). If there is no interest in explicitly forming $\bar{Q}$ and it is acceptable to store the Householder vectors required to first compute the QR factorization of A and next the QR factorization in (2.11), then an approach can be developed for computing the QR factorization of an updated system. The unblocked and blocked algorithm for doing so is given in Figures 2.7 and 2.8, respectively.

Note 4 *Notice that the algorithm is explicitly designed to take advantage of the zeros below the diagonal of R. As a result, factoring A followed by an update of the factorization requires essentially no more computation than the factorization in (2.10). Also, the Householder vectors that are stored below the diagonal are not overwritten. An additional vector b is required to store the "β"s for the unblocked algorithm and an additional matrix is required to store the triangular "T" matrices for the blocked algorithm.*

2.4.10 Solving appended multiple Linear Least-Squares problems

In order to compute multiple Linear Least-Squares solutions, one for each of the systems of linear equations defined by picking one column of the right-hand-side of

$$\begin{pmatrix} A \\ C \end{pmatrix} X = \begin{pmatrix} B \\ D \end{pmatrix}$$

Partition $R \rightarrow \left(\begin{array}{c||c} R_{TL} & R_{TR} \\ \hline\hline R_{BL} & R_{BR} \end{array}\right)$, $C \rightarrow \left(\begin{array}{c||c} C_L & C_R \end{array}\right)$, and $T \rightarrow \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right)$

where R_{TL} and C_L are 0×0 and T_T has 0 rows

while $n(R_{BR}) \neq 0$ **do**

Determine block size k

Repartition

$$\left(\begin{array}{c||c} R_{TL} & R_{TR} \\ \hline\hline R_{BL} & R_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c||c|c} R_{00} & R_{01} & R_{02} \\ \hline\hline R_{10} & R_{11} & R_{12} \\ \hline R_{20} & R_{21} & R_{22} \end{array}\right),$$

$$\left(\begin{array}{c} C_L \\ \hline\hline C_R \end{array}\right) \rightarrow \left(\begin{array}{c||c|c} C_0 & C_1 & C_2 \end{array}\right), \text{ and } \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right) \rightarrow \left(\begin{array}{c} T_0 \\ \hline\hline T_1 \\ \hline T_2 \end{array}\right)$$

where A_{11} is $k \times k$, C_1 has k columns and T_1 has k rows

$$\left[\left(\begin{array}{c} R_{11} \\ \hline C_1 \end{array}\right), \eta, b_1\right] := \left[\left(\begin{array}{c} \{0 \backslash R\}_{11} \\ \hline Y_1 \end{array}\right), \eta, b_1\right] = QR\left(\left(\begin{array}{c} R_{11} \\ \hline C_1 \end{array}\right)\right)$$

Compute T_1 from $\left[\left(\begin{array}{c} I \\ \hline Y_1 \end{array}\right), \eta, b_1\right]$

$$\left(\begin{array}{c} R_{12} \\ \hline C_2 \end{array}\right) := \left(I + \left(\begin{array}{c} I \\ \hline Y_1 \end{array}\right) T_1^T \left(\begin{array}{c|c} I & Y_1^T \end{array}\right)\right) \left(\begin{array}{c} R_{12} \\ \hline C_2 \end{array}\right)$$

Continue with

$$\left(\begin{array}{c||c} R_{TL} & R_{TR} \\ \hline\hline R_{BL} & R_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c||c} R_{00} & R_{01} & R_{02} \\ \hline R_{10} & R_{11} & R_{12} \\ \hline\hline R_{20} & R_{21} & R_{22} \end{array}\right),$$

$$\left(\begin{array}{c||c} C_L & C_R \end{array}\right) \leftarrow \left(\begin{array}{c|c||c} C_0 & C_1 & C_2 \end{array}\right), \text{ and } \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right) \leftarrow \left(\begin{array}{c} T_0 \\ \hline T_1 \\ \hline\hline T_2 \end{array}\right)$$

enddo

Figure 2.8: Blocked update to a QR factorization.

Partition $B \rightarrow \left(\begin{array}{c} B_T \\ \hline\hline B_B \end{array}\right)$, $C \rightarrow \left(\begin{array}{c|\|c} C_L & C_R \end{array}\right)$, and $T \rightarrow \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right)$

where C_L has 0 columns, and B_T and T_T have 0 rows

while $n(C_R) \neq 0$ **do**

Determine block size k

Repartition

$$\left(\begin{array}{c|\|c} C_L & C_R \end{array}\right) \rightarrow \left(\begin{array}{c|c|\|c} C_0 & C_1 & C_2 \end{array}\right),$$

$$\left(\begin{array}{c} B_T \\ \hline\hline B_B \end{array}\right) \rightarrow \left(\begin{array}{c} B_0 \\ \hline\hline B_1 \\ \hline B_2 \end{array}\right), \text{ and } \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right) \rightarrow \left(\begin{array}{c} T_0 \\ \hline\hline T_1 \\ \hline T_2 \end{array}\right)$$

where C_1 has k columns and B_1 and T_1 have k rows

$$\left(\begin{array}{c} B_1 \\ \hline D \end{array}\right) := \left(I + \left(\begin{array}{c} I \\ \hline C_1 \end{array}\right) T_1^T \left(\begin{array}{c|c} I & C_1^T \end{array}\right)\right) \left(\begin{array}{c} B_1 \\ \hline D \end{array}\right)$$

Continue with

$$\left(\begin{array}{c|\|c} C_L & C_R \end{array}\right) \leftarrow \left(\begin{array}{c|c|\|c} C_0 & C_1 & C_2 \end{array}\right),$$

$$\left(\begin{array}{c} B_T \\ \hline\hline B_B \end{array}\right) \leftarrow \left(\begin{array}{c} B_0 \\ \hline B_1 \\ \hline\hline B_2 \end{array}\right), \text{ and } \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right) \leftarrow \left(\begin{array}{c} T_0 \\ \hline T_1 \\ \hline\hline T_2 \end{array}\right)$$

enddo

Figure 2.9: Forward substitution consistent with the QR factorization of an updated matrix.

the following approach will yield the desired result:

- Append $\left(\begin{array}{c|c} A & B \\ C & D \end{array}\right)$.
- Overwrite A with its factors $\{Y \backslash R\}$, also computing matrix T, as in Figure 2.5.
- Overwrite $\begin{pmatrix} R \\ C \end{pmatrix}$ with $\begin{pmatrix} \bar{R} \\ Y^{(C)} \end{pmatrix}$, also computing $T^{(C)}$ as in Figure 2.8.
- Update B by forward substitution as in Figure 2.6.
- Update $\begin{pmatrix} B \\ D \end{pmatrix}$ by forward substitution using the Householder transformations computed as part of the update of R, as in Figure 2.9.
- Solve $\bar{R}X = B_T$ where B_T denotes the top n rows of the updated matrix B.

2.5 Out-of-Core Algorithms

Having now described the in-core algorithm, a similar strategy can be applied to problems that are too large to fit in the available memory of the machine. To deal with these problems, an out-of-core (OOC) algorithm has been developed that allows the bulk of the matrix components to be stored on disk, while only working on select pieces in-core at any one time. The algorithm outlined here is unique in that it is both scalable and efficient.

2.5.1 Out-of-core QR factorization

Traditional OOC algorithms of the QR factorization have used a slab approach, in which the OOC matrix is processed by bringing into memory one or more

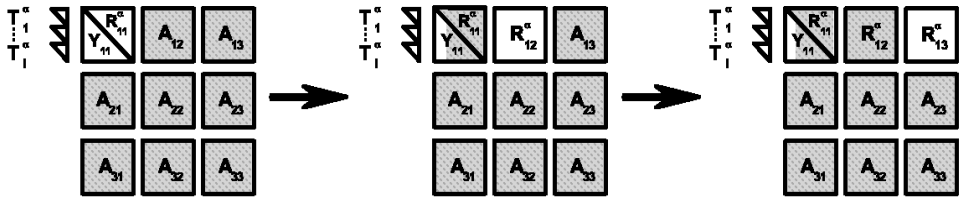
slabs (blocks of columns) of the matrix at a time [9, 38, 12, 65, 50, 66]. The problem with this technique is that it is inherently not scalable in the following sense: As the row dimension, m, of A becomes larger and larger, the width of the slab that can be brought into memory becomes proportionally smaller. As m reaches into the millions, the number of columns able to be brought into memory numbers only in the dozens, even on today's powerful machines with large memories.

The alternative that has been found to the slab approach is to work with the matrix as a collection of tiles, where a tile is a submatrix that is roughly square. As was shown for the OOC Cholesky factorization [65, 45, 44, 28], a tiled approach provides true scalability. It will be shown that the processing of these tiles becomes a simple application of the algorithms in Figures 2.5, 2.6, 2.8, and 2.9. The high performance achieved with these in-core procedures is maintained when processing the tiles, providing the same benefits to the OOC approach. As the problem size increases, additional tiles are simply added to the system, without adversely affecting performance.

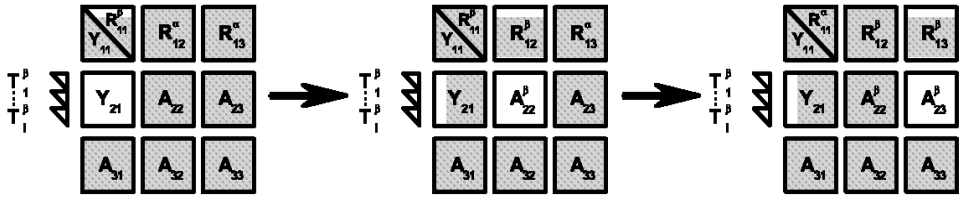
To demonstrate this, the scenario will begin in much the same way as it did with the in-core algorithm, except now the matrix A resides entirely on disk and not in memory. The matrix is partitioned into a series of tiles, as illustrated in Figure 2.10. Note that in that figure, it is the unshaded part of the matrix that is in memory at a typical stage of the algorithm. For the purposes of this example, it is assumed that the matrix A is square and divided into nine tiles of size $t \times t$, forming a 3×3 grid of tiles.

1. The first tile, A_{11}, is read into memory and factored using the in-core

a) A_{11} is factored using the in-core algorithm, then the remaining tiles in the first row are updated using the Householder vectors stored in the lower triangular portion of A_{11}. The Householder vectors are read and applied in narrow panels of width r.



b) The second row of tiles is now processed. A_{21} is factored using the modified in-core algorithm. As in step a, the Householder vectors are applied in panels of width r, while the updates to the first row of tiles are done in horizontal panels of height r. Note the need to create a new set of T matrices.



c) The last tile row is factored, with the first row of tiles receiving its final update. As before, new T matrices are created. The second tile row is not affected.

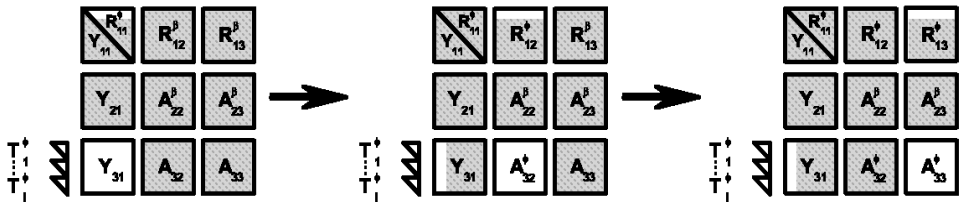


Figure 2.10: Factoring the first row of tiles using the out-of-core approach. Grey regions indicate components that reside on disk.

algorithm in Figure 2.5. Upon completion, A_{11} is written to disk. For now, the "T" matrices created during this step are kept in-core.

Notice that as the dimension t becomes larger, the cost of reading and writing the tiles ($O(t^2)$) improves because it is distributed over the useful computation ($O(t^3)$), i.e., the QR factorization. Consequently, the larger t becomes, the less significant the I/O overhead. This is why the tile size is encouraged to be as large as will fit into memory.

2. Next, tile A_{12} is brought into memory. It is updated consistent with the factorization of A_{11}, using the Householder vectors that have overwritten the lower triangular part of A_{11} and the "T" matrices still in memory. In other words, the algorithm in Figure 2.6 is employed. Once updated, A_{12} is written back to disk.

   On the surface, it would thus appear that two tiles must be in memory, consequently limiting the tile dimension t. However, a closer look at the update in the body of the loop of the algorithm in Figure 2.6 shows that only a panel of columns of A_{11} needs to be brought into memory, which can be discarded as soon as it has been used to update A_{12}. Thus, at most $t \times k$ elements of A_{11} need to be in memory at a time. The cost of bringing these elements into memory is distributed over $O(kt^2)$ computations.

   The remaining tiles in the first row are processed similarly. Once the entire first row has been processed, the "T" matrices computed in the factorization of A_{11} can be written to disk.

3. After processing the first row, A_{21} is brought into memory. It must be updated together with A_{11} according to the algorithm in Figure 2.8,

generating a new set of "T" matrices. Once updated, A_{21} is written to disk, while the newly generated "T" matrices are kept in memory.

Again, it would appear that two tiles must be in memory, thus limiting the tile dimension t. However, the update in the body of the loop of the algorithm in Figure 2.8 shows that only a panel of rows of A_{11} needs to be brought into memory, which can be written back to disk as soon as it has been used to update A_{21}. Thus, at most $k \times t$ elements of A_{11} need to be in memory at a time. Again, the cost of bringing these elements into memory is distributed over $O(kt^2)$ computations.

4. Once A_{21} is updated, A_{22} is brought into memory, to be updated according to Step 3 above, using the algorithm in Figure 2.9.

   It would appear that now A_{21}, A_{12}, and A_{22} must all be in memory simultaneously. However, the update in the body of the loop in Figure 2.9 requires only a panel of columns of A_{21} and a panel of rows of A_{12} to be in memory. Thus only A_{22} needs to be kept in memory, while panels of the other two matrices are streamed from disk. The cost of the I/O involved is $O(kt)$ per iteration of the loop, which is distributed over $O(kt^2)$ computations.

   The remaining tiles in the second row are processed similarly.

5. The third row of tiles is handled in the same manner as described in Steps 3 and 4. A_{31} is first factored with A_{11}, creating a new set of "T" matrices and overwriting A_{31} with the corresponding Householder vectors. A_{32} is brought into memory and updated with column panels from A_{31}, while also updating A_{12} in row panels of height k. The same is done for A_{33}

and A_{13}. Note that only the first and third row of tiles are affected by these operations.

6. Now, Steps 1–5 start to repeat: A_{22} is factored as A_{11} was in Step 1. The remaining tiles in the second row are processed as described in Step 2. The tiles in the third row below the tiles on the diagonal are processed as in Step 3, and the remaining tiles in the third row are processed as in Steps 4 and 5. After this, it is back to Step 1 with A_{33} and so forth.

A detailed description for this algorithm is provided in Figure 2.11.

Note 5 *In principle, most of the memory can be dedicated to storing a single $t \times t$ tile. This allows t to be as large as possible, which then improves the indicated ratios of I/O to useful computation.*

2.5.2 Solving multiple linear least-squares problems

The appended matrix, $\left(\begin{array}{c|c} A & B \end{array} \right)$, seen earlier for the multiple least-squares problem can be accommodated using the technique outlined in Figure 2.12. As before, the T matrices were stored when A was factored, so they do not need to be computed again.

Note 6 *It is possible that the number of columns in B is greater than the tile size, t, explaining why B is described as columns of tiles.*

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$ and $T \rightarrow \left(\begin{array}{c} T_T \\ \hline T_B \end{array}\right)$

where A_{TL} is 0×0 and T_T has 0 rows

while $n(A_{BR}) \neq 0$ **do**

Determine block size t

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline A_{10} & A_{11} & A_{12} \\ \hline A_{20} & A_{21} & A_{22} \end{array}\right) \text{ and } \left(\begin{array}{c} T_T \\ \hline T_B \end{array}\right) \rightarrow \left(\begin{array}{c} T_0 \\ \hline T_1 \\ \hline T_2 \end{array}\right)$$

where

A_{11} is $t \times t$

A_{12} is stored as a series of j, $t \times t$ tiles: $A_{12} \leftarrow \left(\begin{array}{c|c|c} J_1 & \cdots & J_j \end{array}\right)$

A_{21} is stored as a series of i, $t \times t$ tiles: $A_{21} \leftarrow \left(\begin{array}{c} K_1 \\ \hline \vdots \\ \hline K_i \end{array}\right)$

A_{22} is stored as a $i \times j$ grid of $t \times t$ tiles: $A_{21} \leftarrow \left(\begin{array}{c|c|c} G_{11} & \cdots & G_{1j} \\ \hline \vdots & \ddots & \vdots \\ \hline G_{i1} & \cdots & G_{ij} \end{array}\right)$

T_1 has $(i+1)$ elements, each with t rows: $T_1 \leftarrow \left(\begin{array}{c} T_{01} \\ \hline \vdots \\ \hline T_{i1} \end{array}\right)$

$[A_{11}, b_1] := [\{Y \backslash R\}_{11}, b_1] = QR(A_{11})$

Compute T_{01} from $[\{Y \backslash R\}_{11}, b_1]$

for $p = 1, j$

$J_p := \left(I + Y_{11} T_{01}^T Y_{11}^T\right)(J_p)$

end

for $m = 1, i$

$\left[\left(\begin{array}{c} R_{11} \\ \hline K_m \end{array}\right), b_1\right] := \left[\left(\begin{array}{c} \{0 \backslash R\}_{11} \\ \hline Y_m \end{array}\right), b_1\right] = QR\left(\left(\begin{array}{c} R_{11} \\ \hline K_m \end{array}\right)\right)$

Compute T_{m1} from $\left[\left(\begin{array}{c} I \\ \hline Y_m \end{array}\right), b_1\right]$

for $n = 1, j$

$\left(\begin{array}{c} J_n \\ \hline G_{mn} \end{array}\right) := \left(I + \left(\begin{array}{c} I \\ \hline Y_m \end{array}\right) T_{m1}^T \left(\begin{array}{c|c} I & Y_m^T \end{array}\right)\right) \left(\begin{array}{c} J_n \\ \hline G_{mn} \end{array}\right)$

end

end

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline A_{10} & A_{11} & A_{12} \\ \hline A_{20} & A_{21} & A_{22} \end{array}\right) \text{ and } \left(\begin{array}{c} T_T \\ \hline T_B \end{array}\right) \leftarrow \left(\begin{array}{c} T_0 \\ \hline T_1 \\ \hline T_2 \end{array}\right)$$

enddo

Figure 2.11: Out-of-core Householder QR factorization.

Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline\hline A_{BL} & A_{BR} \end{array}\right)$, $B \rightarrow \left(\begin{array}{c} B_T \\ \hline\hline B_B \end{array}\right)$ and $T \rightarrow \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right)$

where A_{TL} is 0×0 and B_T and T_T have 0 rows

while $n(A_{BR}) \neq 0$ **do**

Determine block size t

Repartition

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline\hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline\hline A_{10} & A_{11} & A_{12} \\ \hline\hline A_{20} & A_{21} & A_{22} \end{array}\right),$$

$$\left(\begin{array}{c} B_T \\ \hline\hline B_B \end{array}\right) \rightarrow \left(\begin{array}{c} B_0 \\ \hline\hline B_1 \\ \hline\hline B_2 \end{array}\right), \text{ and } \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right) \rightarrow \left(\begin{array}{c} T_0 \\ \hline\hline T_1 \\ \hline\hline T_2 \end{array}\right)$$

where

A_{11} is $t \times t$,

B_1 is stored as a series of j, $t \times t$ tiles: $B_1 \leftarrow \left(\begin{array}{c|c|c} E_1 & \cdots & E_j \end{array}\right)$

B_2 is stored as a $i \times j$ grid of $t \times t$ tiles: $B_2 \leftarrow \left(\begin{array}{c|c|c} F_{11} & \cdots & F_{1j} \\ \hline \vdots & \ddots & \vdots \\ \hline F_{i1} & \cdots & F_{ij} \end{array}\right)$

T_1 has $(i+1)$ elements, each with t rows: $T_1 \leftarrow \left(\begin{array}{c} T_{01} \\ \hline \vdots \\ \hline T_{i1} \end{array}\right)$

---

for $p = 1, j$

$E_p := \left(I + Y_{11} T_{01}^T Y_{11}^T\right)(E_p)$

end

for $m = 1, i$

for $n = 1, j$

$$\left(\begin{array}{c} E_n \\ \hline F_{mn} \end{array}\right) := \left(I + \left(\begin{array}{c} I \\ \hline Y_m \end{array}\right) T_{m1}^T \left(\begin{array}{c|c} I & Y_m^T \end{array}\right)\right) \left(\begin{array}{c} E_n \\ \hline F_{mn} \end{array}\right)$$

end

end

Continue with

$$\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline\hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline\hline A_{10} & A_{11} & A_{12} \\ \hline\hline A_{20} & A_{21} & A_{22} \end{array}\right),$$

$$\left(\begin{array}{c} B_T \\ \hline\hline B_B \end{array}\right) \leftarrow \left(\begin{array}{c} B_0 \\ \hline\hline B_1 \\ \hline\hline B_2 \end{array}\right), \text{ and } \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right) \leftarrow \left(\begin{array}{c} T_0 \\ \hline\hline T_1 \\ \hline\hline T_2 \end{array}\right)$$

enddo

Figure 2.12: Out-of-core forward substitution-like of right-hand-side matrix B.

2.5.3 Out-of-core updating

To update an existing OOC solution with a new set of equations is straightforward, as the OOC algorithm was designed to handle problems of arbitrary length. As Figure 2.13 describes, the new data is simply divided into the appropriate tiles and combined with the existing solution in the same manner that the 2nd and 3rd rows of tiles were handled in the previous section.

2.5.4 Solving multiple appended linear least-squares problems

Computing the multiple least-squares solution is analogous to the method outlined in section 2.4.10

- Append $\left(\begin{array}{c|c} A & B \\ C & D \end{array}\right)$.
- Overwrite A with its factors $\{Y \backslash R\}$, also computing matrix T, as in Fig. 2.11.
- Overwrite $\begin{pmatrix} R \\ C \end{pmatrix}$ with $\begin{pmatrix} \bar{R} \\ Y^{(C)} \end{pmatrix}$, also computing $T^{(C)}$ as in Fig. 2.13.
- Update B by forward substitution as in Fig. 2.12.
- Update $\begin{pmatrix} B \\ D \end{pmatrix}$ by forward substitution using the Householder transformations computed as part of the update of R, as in Fig. 2.14.
- Solve $\bar{R}X = B_T$ where B_T denotes the top n rows of the updated matrix B, as in Fig. 2.15.

Partition $R \rightarrow \left(\begin{array}{c||c} R_{TL} & R_{TR} \\ \hline\hline R_{BL} & R_{BR} \end{array}\right)$, $C \rightarrow \left(\begin{array}{c||c} C_L & C_R \end{array}\right)$, and $T \rightarrow \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right)$

where R_{TL} and C_L are 0×0 and T_T has 0 rows

while $n(R_{BR}) \neq 0$ **do**

Determine block size t

Repartition

$$\left(\begin{array}{c||c} R_{TL} & R_{TR} \\ \hline\hline R_{BL} & R_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c||c|c} R_{00} & R_{01} & R_{02} \\ \hline\hline R_{10} & R_{11} & R_{12} \\ \hline\hline R_{20} & R_{21} & R_{22} \end{array}\right),$$

$$\left(\begin{array}{c} C_L \\ \hline\hline C_R \end{array}\right) \rightarrow \left(\begin{array}{c||c|c} C_0 & C_1 & C_2 \end{array}\right), \text{ and } \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right) \rightarrow \left(\begin{array}{c} T_0 \\ \hline\hline T_1 \\ \hline\hline T_2 \end{array}\right)$$

where

R_{11} is $t \times t$

R_{12} is stored as a series of j, $t \times t$ tiles: $R_{12} \leftarrow \left(\begin{array}{c|c|c} W_1 & \cdots & W_j \end{array}\right)$

C_1 is stored as a series of i, $t \times t$ tiles: $C_1 \leftarrow \left(\begin{array}{c} V_1 \\ \hline \vdots \\ \hline V_i \end{array}\right)$

C_2 is stored as a $i \times j$ grid of $t \times t$ tiles: $C_2 \leftarrow \left(\begin{array}{c|c|c} U_{11} & \cdots & U_{1j} \\ \hline \vdots & \ddots & \vdots \\ \hline U_{i1} & \cdots & U_{ij} \end{array}\right)$

T_1 has i elements, each with t rows: $T_1 \leftarrow \left(\begin{array}{c} T_{11} \\ \hline \vdots \\ \hline T_{i1} \end{array}\right)$

---

for $m = 1, i$

$$\left[\left(\begin{array}{c} R_{11} \\ \hline V_m \end{array}\right), b_1\right] := \left[\left(\begin{array}{c} \{0 \backslash R\}_{11} \\ \hline Y_m \end{array}\right), b_1\right] = QR\left(\left(\begin{array}{c} R_{11} \\ \hline V_m \end{array}\right)\right)$$

Compute T_{m1} from $\left[\left(\begin{array}{c} I \\ \hline Y_m \end{array}\right), b_1\right]$

for $n = 1, j$

$$\left(\begin{array}{c} W_n \\ \hline U_{mn} \end{array}\right) := \left(I + \left(\begin{array}{c} I \\ \hline Y_m \end{array}\right) T_{m1}^T \left(\begin{array}{c|c} I & Y_m^T \end{array}\right)\right) \left(\begin{array}{c} W_n \\ \hline U_{mn} \end{array}\right)$$

end

end

Continue with

$$\left(\begin{array}{c||c} R_{TL} & R_{TR} \\ \hline\hline R_{BL} & R_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c||c} R_{00} & R_{01} & R_{02} \\ \hline R_{10} & R_{11} & R_{12} \\ \hline\hline R_{20} & R_{21} & R_{22} \end{array}\right),$$

$$\left(\begin{array}{c||c} C_L & C_R \end{array}\right) \leftarrow \left(\begin{array}{c|c||c} C_0 & C_1 & C_2 \end{array}\right), \text{ and } \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right) \leftarrow \left(\begin{array}{c} T_0 \\ \hline T_1 \\ \hline\hline T_2 \end{array}\right)$$

enddo

Figure 2.13: Update Using Out-of-core QR factorization.

Partition $B \rightarrow \left(\begin{array}{c} B_T \\ \hline\hline B_B \end{array}\right)$, $C \rightarrow \left(\begin{array}{c|c} C_L & C_R \end{array}\right)$, and $T \rightarrow \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right)$

where C_L has 0 columns, and B_T and T_T have 0 rows

while $n(C_R) \neq 0$ **do**

Determine block size t

Repartition

$$\left(\begin{array}{c|c} C_L & C_R \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} C_0 & C_1 & C_2 \end{array}\right),$$
$$\left(\begin{array}{c} B_T \\ \hline\hline B_B \end{array}\right) \rightarrow \left(\begin{array}{c} B_0 \\ \hline\hline B_1 \\ \hline B_2 \end{array}\right), \text{ and } \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right) \rightarrow \left(\begin{array}{c} T_0 \\ \hline\hline T_1 \\ \hline T_2 \end{array}\right)$$

where

B_1 is stored as a series of j, $t \times t$ tiles: $B_1 \leftarrow \left(\begin{array}{c|c|c} E_1 & \cdots & E_j \end{array}\right)$

C_1 is stored as a series of i, $t \times t$ tiles: $C_1 \leftarrow \left(\begin{array}{c} V_1 \\ \hline \vdots \\ \hline V_i \end{array}\right)$

D is stored as a $i \times j$ grid of $t \times t$ tiles: $D \leftarrow \left(\begin{array}{c|c|c} P_{11} & \cdots & P_{1j} \\ \hline \vdots & \ddots & \vdots \\ \hline P_{i1} & \cdots & P_{ij} \end{array}\right)$

T_1 has i elements, each with t rows: $T_1 \leftarrow \left(\begin{array}{c} T_{11} \\ \hline \vdots \\ \hline T_{i1} \end{array}\right)$

---

for $m = 1, i$

for $n = 1, j$

$$\left(\begin{array}{c} E_n \\ \hline P_{mn} \end{array}\right) := \left(I + \left(\begin{array}{c} I \\ \hline V_m \end{array}\right) T_{m1}^T \left(\begin{array}{c|c} I & V_m^T \end{array}\right)\right) \left(\begin{array}{c} E_n \\ \hline P_{mn} \end{array}\right)$$

end

end

Continue with

$$\left(\begin{array}{c|c} C_L & C_R \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} C_0 & C_1 & C_2 \end{array}\right),$$
$$\left(\begin{array}{c} B_T \\ \hline\hline B_B \end{array}\right) \leftarrow \left(\begin{array}{c} B_0 \\ \hline\hline B_1 \\ \hline\hline B_2 \end{array}\right), \text{ and } \left(\begin{array}{c} T_T \\ \hline\hline T_B \end{array}\right) \leftarrow \left(\begin{array}{c} T_0 \\ \hline\hline T_1 \\ \hline\hline T_2 \end{array}\right)$$

enddo

Figure 2.14: Out-of-core forward substitution consistent with the Out-of-core QR factorization of an updated matrix.

Partition $R \rightarrow \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline\hline R_{BL} & R_{BR} \end{array}\right)$, $B \rightarrow \left(\begin{array}{c} B_T \\ \hline\hline B_B \end{array}\right)$, and $X \rightarrow \left(\begin{array}{c} X_T \\ \hline\hline X_B \end{array}\right)$

where R_{TL}, B_T and X_T are 0×0

while $n(R_{BR}) \neq 0$ **do**

Determine block size t

Repartition

$$\left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline\hline R_{BL} & R_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} R_{00} & R_{01} & R_{02} \\ \hline R_{10} & R_{11} & R_{12} \\ \hline\hline R_{20} & R_{21} & R_{22} \end{array}\right),$$

$$\left(\begin{array}{c} B_T \\ \hline\hline B_B \end{array}\right) \rightarrow \left(\begin{array}{c} B_0 \\ \hline B_1 \\ \hline\hline B_2 \end{array}\right),$$

$$\left(\begin{array}{c} X_T \\ \hline\hline X_B \end{array}\right) \rightarrow \left(\begin{array}{c} X_0 \\ \hline X_1 \\ \hline\hline X_2 \end{array}\right)$$

where

R_{11} is $t \times t$

B_1 has t rows

B_2 has i elements, each sized $t \times 1$: $B_2 \leftarrow \left(\begin{array}{c} L_1 \\ \hline \vdots \\ \hline L_i \end{array}\right)$

R_{12} is stored as a series of i, $t \times t$ tiles: $R_{12} \leftarrow \left(\begin{array}{c|c|c} W_1 & \cdots & W_i \end{array}\right)$

---

for $m = 1, i$

$B_1 := B_1 - W_m L_m$

end

$X_1 =$ in-core back substitution of R_{11} and B_1

Continue with

$$\left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline\hline R_{BL} & R_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} R_{00} & R_{01} & R_{02} \\ \hline\hline R_{10} & R_{11} & R_{12} \\ \hline R_{20} & R_{21} & R_{22} \end{array}\right),$$

$$\left(\begin{array}{c} B_T \\ \hline\hline B_B \end{array}\right) \leftarrow \left(\begin{array}{c} B_0 \\ \hline\hline B_1 \\ \hline B_2 \end{array}\right),$$

$$\left(\begin{array}{c} X_T \\ \hline\hline X_B \end{array}\right) \leftarrow \left(\begin{array}{c} X_0 \\ \hline\hline X_1 \\ \hline X_2 \end{array}\right)$$

enddo

Figure 2.15: Out-of-core backward substitution.

2.5.5 Implementation

It is well-known that a scalable implementation of dense linear algebra operations on distributed memory architecture requires the use of a so-called two-dimensional matrix distribution [32, 56]. Moreover, to ensure load-balance as the active part of the matrix shrinks, an overdecomposition and wrapping of the matrix is typically employed [41, 57, 67, 17].

The observation now is that if one has parallel implementations of the algorithms in Figures 2.4–2.9, then the parallel implementation of the OOC algorithm becomes straight-forward. The parallel implementation of the QR factorization is well-understood, and is available as part of the PLAPACK package, as well as part of the Scalable Linear Algebra Package (ScaLAPACK) [7]. Since the remaining algorithms are, in essence, merely variations of the QR factorization, they have been implemented as modifications of the PLAPACK QR factorization. Further modifications had to be made so that the panels being streamed from disk were read into memory and/or written out to disk at the appropriate time. To facilitate these operations, the Parallel Out-of-Core Linear Algebra Package (POOCLAPACK) [68, 31] was developed as the OOC extension to PLAPACK. Finally, a routine that manages the processing of the tiles was also written.

2.5.6 Optimizing I/O performance

The OOC method described in Section 2.5.1 advocates a single-tile method, in which most of memory is dedicated to a single tile. As argued, this is desirable because the I/O overhead decreases as the tile size increases. While this is easy to justify theoretically, in this section a practical consideration is

pointed out which suggests that keeping two tiles in memory may lead to better performance. The two tile approach also leads to a simpler implementation.

First, a few details about the storage of matrices. The matrices assigned locally to each processor as part of tiles are stored in memory in column-major order. Similarly, on disk, they are stored in column-major order. More precisely, the columns are stored so that if a panel of columns is read by a processor from disk, they are all contiguous in memory. This makes the reading of a tile and of panels of columns relatively cheap, since I/O carries a large start-up cost (latency). In other words, a panel of columns can be read essentially at peak bandwidth. By contrast, the reading of a panel of rows is generally staged as the reading of individual columns of that panel, incurring a latency related cost for each such column. This makes the reading of a panel of rows prohibitively expensive.

The update in Step 2 in Section 2.5.1 requires only column panels (of Householder vectors) to be read from disk. By contrast, the operations in Steps 3 and 4 require panels of rows to be brought in. Thus, it becomes advantageous to bring the entire tile from which panels of rows are to be used into memory, leading to a two-tile OOC algorithm. It is this approach that was actually implemented and used to obtain the performance numbers described in the next section.

Note 7 *By transposing tiles above the diagonal after processing, it is possible to implement the one-tile approach while still only reading panels of columns. This was not done in an effort to keep the implementation simple.*

2.6 Performance

In this section, the presented algorithm is shown to attain very high performance on distributed memory parallel architectures.

2.6.1 Target architectures

The POOCLAPACK implementation of the OOC QR factorization and update algorithm is portable essentially to any platform that supports the Message-Passing Interface [23, 55] and the Basic Linear Algebra Subprograms [39, 14, 13]. To date, the implementation has been ported to the SGI Origin 3000 and Linux PC cluster environments, in addition to the Cray T3E and IBM P-series systems.

Performance numbers were recorded on two different architectures:

- The Cray T3E-600. The system on which the experiments were performed has 272 total processors, each with 128MB of available memory. The T3E operates at a peak theoretical performance of 600 millions of floating point operations per second per processor (MFLOPS/sec/proc). For reference, the matrix-matrix multiply operation (DGEMM) was benchmarked at 445 MFLOPS/sec/proc for the particular machine used in this study. The BLAS used was provided as part of the Cray Scientific Library. It should also be noted that since the T3E is a true 64-bit platform, all arithmetic was done using 64-bit precision.

- The IBM P690. The system on which the experiments were performed consists of SMP nodes, where each node consists of 16 Power4 (1.3 GHz) processors, with 32 GBytes of available memory. The P690s operate

at four FLOPS per cycle for a peak theoretical performance of 5200 MFLOPS/sec/proc, with a DGEMM benchmark of 3723 MFLOPS/sec/proc. IBM's optimized Engineering and Scientific Subroutine Library (ESSL) was used in place of the standard BLAS library. Again, all computation was performed in 64-bit arithmetic.

2.6.2 Reporting performance

The operation count for a Householder transform based QR factorization of an $m \times m$ matrix is given by approximately $\frac{4}{3}m^3$ floating point operations. While the OOC algorithm requires more operations, due to the accumulation and application of the "T" matrices, it is this operation count that represents the *useful* computation.

Thus, given $T_p(m)$, the time in seconds required on p processors to factor an $m \times m$ matrix, the rate in MFLOPS/sec/proc at which the processors compute is given by the formula

$$R_p(m) = \frac{\frac{4}{3}m^3}{T_p(m)} \times \frac{10^{-6}}{p}$$

Now, since the bulk of the computation is cast in terms of local matrix-matrix multiplications, the upper bound on $R_p(m)$ is given by the rate in MFLOPS/sec attained by BLAS routine DGEMM[13], which shall be denoted by R_{dgemm}. This is generally considered to be the peak performance that can be attained per processor, or the "realizable" peak of the system. It should be noted that the QR factorization will never actually attain this realizable peak due to the added complexity of the operations involved, but given an increasing amount of memory and problem size, it would approach it in the limit. Therefore, the

performance of the OOC implementation will be reported as a percentage of the realizable peak by the ratio

$$\frac{R_p(m)}{R_{dgemm}}.$$

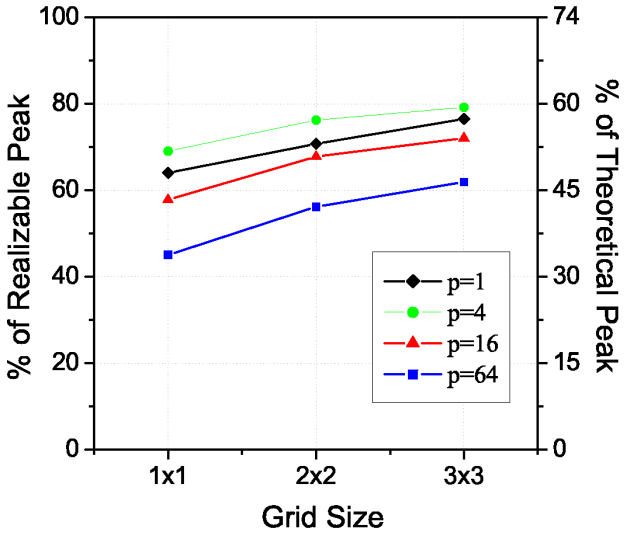
Depending on the architecture, this realizable peak is 70-99% of the theoretical peak of the processor, which is defined by the clock speed multiplied by the number of floating point operations that can be performed per clock cycle. Reporting performance relative to the realizable peak is intended to give a clearer insight into the overhead incurred by the parts of the QR factorization that are not cast in terms of matrix-matrix multiplication, the overhead due to the parallelization, and the overhead due to I/O.

2.6.3 Results

Figure 2.16 illustrates the performance of the in-core and OOC QR factorization algorithms. In our experiment, both the number of processors used and the problem size are varied in the following way: Parameter t is chosen as the dimension of the tiles that will be kept in memory. Naturally, as the number of processors increases, the total available memory increases, and t can be increased. Factorizations of problems of size 1×1 tiles through 3×3 tiles were subsequently timed (reported as the Grid Size along the x-axis). The curves connect the data points corresponding to the number of processors indicated in the legend.

The columns and lines of the 1×1 case represent the performance of the in-core algorithm, since only one tile is involved. The other cases involving multiple tiles were all computed using the OOC algorithm. As the figure shows,

T3E Performance



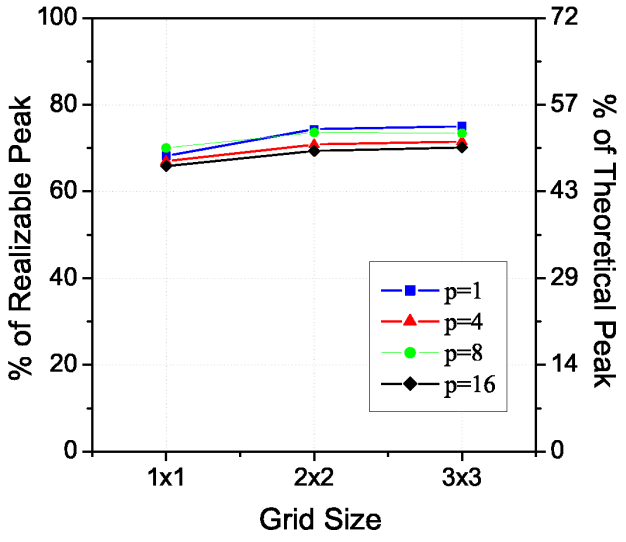
Problem Size

# Procs	1	4	8	16
Tile Size	2088	4704	8448	18432
n (1 x 1)	2088	4704	8448	18432
n (2 x 2)	4176	9408	16896	36864
n (3 x 3)	6264	14112	25344	55296

Compute Time (sec.)

# Procs	1	4	8	16
n (1 x 1)	43	112	195	651
n (2 x 2)	308	818	1332	4176
n (3 x 3)	962	2657	4231	12776

IBM Performance



Problem Size

# Procs	1	4	8	16
Tile Size	7900	21000	30000	42000
n (1 x 1)	7900	21000	30000	42000
n (2 x 2)	15800	42000	60000	84000
n (3 x 3)	23700	63000	90000	126000

Compute Time (sec.)

# Procs	1	4	8	16
n (1 x 1)	259	1238	1726	2518
n (2 x 2)	1900	9365	13130	19128
n (3 x 3)	6349	31286	44439	63770

Figure 2.16: Performance of the OOC algorithm on a Cray T3E and IBM P690.

the performance is quite respectable for an operation as complex as the QR factorization, achieving 60-80% of the realizable peak for most cases.

It is interesting to note that as the problem size becomes larger, the performance improves. This can be explained by the fact that as the problem size increases, more of the computation is in the operations in Figures 2.6 and 2.9. This casts more of the computation in matrix-matrix multiplication, the operation that attains the highest performance.

There is a noticeable difference between the two architectures regarding scalability as the number of processors is increased. This can largely be attributed to the fact that the I/O performance of the specific Cray T3E used for these experiments becomes a bottleneck as more processors access the disk simultaneously.

2.6.4 Further possible improvements

The use of asynchronous I/O (i.e., overlapping I/O with computation) was explored in a previous study on parallel OOC implementation of the Cholesky factorization [45, 44, 28]. While it was determined that a slight performance increase could be achieved on machines with slower I/O bandwidths, the complexity of the code required to do this was considered prohibitive for the algorithms presented here. Advances in I/O technology with newer high performance machines also render this performance increase practically negligible. Consequently, asynchronous I/O was not used to achieve the performance numbers described in Figure 2.16.

While the results above were obtained using the Cray T3E and IBM P690, it should be noted that the performance of the algorithm on other plat-

forms is comparable when examining the speed as a percentage of the realizable peak.

2.7 Conclusion

In this chapter, a software tool has been described whose main purpose is to satisfy the computational demands and requirements of the GRACE mission. Using the PLAPACK and POOCLAPACK libraries as building blocks, a suite of in-core and out-of-core algorithms were developed that can compute the parallel QR factorization of dense matrices that are nearly arbitrary in size.

The in-core algorithms were eventually combined into the Advanced Equation Solver for Parallel Systems (AESoP), a processing tool designed to estimate high degree and order gravity field models. The implementation of AESoP is both flexible and portable, and has been designed such that future enhancements can be easily adapted. AESoP now serves as an integral part of the GRACE processing stream, and has evolved over the years to incorporate the functionality needed to fully explore the GRACE data.

The results of Section 2.5 demonstrated that a modification of the standard in-core QR factorization algorithm, combined with a tile-based approach for out-of-core implementations, results in a highly efficient and powerful method for computing QR factorizations of large, dense matrices. The implementation is unique in that it is scalable both as the number of processors is increased and as the problem size is increased (for a fixed number of processors). The performance of these algorithms is also impressive, reaching roughly 80% of the "realizable" peak in some cases. The application of the out-of-core algorithms has already demonstrated its value through the prelim-

inary solution of a rigorous (i.e., without the use of approximation techniques) 360x360 solution [10], complete with a full covariance. This solution, involving the estimation of over 130,000 parameters and well over a terabyte of data, was the largest rigorously computed gravity field ever created.

While the development of these algorithms was guided by the need to solve for complex gravity models, the application of the concepts presented here are not limited to the GRACE mission. As mentioned before, the libraries necessary to create such routines are widely available and are compatible with a number of different platforms. Any problem requiring the least squares solution of a large dense linear system could easily adapt the algorithms discussed in this chapter to achieve similar results. The tile-based approach, in particular, is well suited for other types of dense linear algebra operations, such as the Cholesky decomposition [28, 45, 44] and the LU factorization [27].

Chapter 3

Errors of Omission and Commission

3.1 Introduction

The goal of this chapter is to study the influence that certain processing choices have on the quality of the GRACE gravity field models. When deciding how to combine the various data sources to achieve the most accurate estimates of the gravity coefficients, a number of choices must be made. These include matters such as which reference field to use, the resolution of this force model, and how far to extend the spherical harmonics for a given data type. These choices all have an inherent level of error associated with them that are typically referred to as errors of omission and commission.

One example of the errors of omission involves the fact that, at some point, the geopotential function (see Appendix A.1) must be discretized, or truncated. The point at which this infinite series is truncated depends on many factors, such as the observability of the parameters involved or the limitations of the available computational resources. Regardless of where the potential function is truncated, there will always be a level of error associated with the fact that certain parameters have been left out of the solution process.

Additional errors of omission and commission arise from the limitations in the reference, or nominal, model used in the estimation process (see Appendix A.2). The reference field used in the batch estimation procedure often

represents the best current knowledge of the Earth's gravity field. Nonetheless, the variations of the Earth's mass and density are not known perfectly, resulting in errors in the nominal field. These imperfections in the reference model are classified as errors of commission. The resolution of the reference field is also finite, meaning that there are some forces that get left untreated. These unmodeled parameters, however small, result in another form of omission error.

It should be emphasized that the errors of omission and commission are not mutually exclusive, and are often correlated. A typical gravity solution will contain varying types of these error sources, as described in the following example. Suppose we wish to generate a gravity field model out to spherical harmonic degree and order 160. Assume the sampling rate of the measurement data is sufficient to remove concerns about observability of these parameters. When the problem is linearized, the measurements partials are evaluated with a reference, or nominal, field (see Appendix A.2 for details). If this nominal field is only available out to degree and order 360, the unmodeled coefficients beyond 360 would introduce some degree of omission error in the ensuing solution. The nominal field used is not a perfect model of the Earth's gravity field, so the commission errors represent the effect that imperfections in the model have on the parameters that are estimated. Finally, another form of omission error is introduced by the fact that the measurement partials extend only to degree and order 160. Even if modeling errors were not a concern, there will always be a certain amount of omission error associated with the fact that we are trying to estimate an infinite series with a finite number of parameters.

The use of simulations is an essential step towards understanding the nature of omission and commission errors. When working with real data, it is

nearly impossible to distinguish these errors from the true signal, or from any other error sources that may be present. In a controlled environment, however, the truth is known and can be used to accurately measure the impact of any errors in the system. The conclusions of this chapter were based on a series of simulated experiments designed to quantify the omission and commission errors in the context of the GRACE RL01 [1] processing scenario.

3.2 Simulation Details

Much of the theory, development and application of the simulations conducted in this study followed directly from earlier work done by Kim [37]. The reader is encouraged to read Kim's work for a much more in-depth explanation of the measurement noise modeling and parameterization choices employed in this study.

3.3 Simulation Parameters

The simulations were designed to emulate the processing environment surrounding the first release of the GRACE data (RL01), making use of the same parameterizations and modeling assumptions as those used to create GGM01C [60].

The orbit configuration for the simulations starts with the two GRACE satellites at an altitude of approximately 465 km, a separation angle of 2 degrees ($\sim$ 240 km), and an inclination of 89 degrees. The period for this orbit is roughly 5600 seconds. Each of the simulations covered a 30 day time span,

[1] Data processing standards and a user handbook for the GRACE gravity solutions are available at `www.csr.utexas.edu/grace/publications/handbook`

with the measurement partials created using a one day orbit integration period (i.e., one day arcs). To avoid any sampling errors, the repeat orbit period was designed to be greater than 30 days.

Note 8 *The details of the simulation process, including a step-by-step procedure and a more complete discussion of the errors introduced to the system, can be found in Appendix C.*

The simulations emulated the RL01 processing by relying on the same primary observables as the GRACE mission: GPS double-differenced (GPSDD) observations and K-band inter-satellite range-rate observations. The GPSDD observations were created from a 24 satellite GPS constellation and corresponding 6 station ground network. The observations were sampled at 60 second intervals and included a 1 cm double-difference white noise error. The K-band range-rate (KBR) data was sampled at 10 second intervals and included a standard set of measurement error sources, such as system, oscillator and multipath noise. Additional disturbances such as atmospheric drag, solar radiation pressure and Earth radiation pressure were also applied to the system.

Table 3.1 outlines the parameterization used to achieve the results described in this chapter. This table also highlights a few departures from the simulations used in Kim's studies. In order to account for various mismodeled forces acting on the satellites, Kim made use of a set of constant tangential (CT) parameters that were implemented directly into the dynamical equations (i.e., dynamic empirical parameters). For GRACE RL01 processing, these parameters were not included in the estimation process because it was assumed that any mismodeled forces would be accounted for by the accelerometers. To

Parameter Type	Abbreviation	Duration
Accelerometer bias	AC0	1 day
Accelerometer scale	AC1	30 days
GPS DD Ambiguities	DD AMB	Per cycle slip
Low-low bias	LLB	45 min
Low-low bias periodic	LLBP	90 min
Low-low bias rate	LLBD	45 min
Initial conditions	IC	1 day
Gravity coefficients	GEO	30 days

Table 3.1: Parameterization used in the simulations.

keep consistent with the RL01 processing, the simulations did not use CT parameters, instead relying on the modeled accelerometer measurements and a set of accelerometer biases and scale factors. Another notable difference was the exclusion of the low-low bias periodic rate (LLBPD) terms, one of the kinematic empirical parameters outlined by Kim to account for errors in the inter-satellite ranging measurements. The use of these terms were found to make little or no contribution to the RL01 processing, and were subsequently removed from these simulations as well.

Modeling errors were introduced to some of the simulations through the use of a clone field, or a reference field whose coefficients have been intentionally corrupted from the truth with noise. The modeling errors for this study were created by applying a 1-σ error variation to the coefficients of the truth reference field (see Section C.1 for details). By creating the errors in this manner, the difference in the coefficients between the clone and truth field stay within one standard deviation of the truth, i.e., within the uncertainty of the truth field. This clone field was then used to evaluate the measurement partials in any simulation that required commission errors in the nominal field.

3.4 Truncation Errors

The first series of errors to be examined were the omission errors associated with truncating the measurement partials of the KBR and GPS data. To isolate this error, a series of experiments were conducted in which the only variant was the degree and order to which the KBR or GPS partials were extended. To avoid the introduction of modeling errors into the system, the same truth reference field that was used to generate the simulated observations was also used to evaluate the partials for these experiments. The force model resolution (FMR) was also identical to that used in the observation generation phase, spanning to degree and order 360, ensuring that no modeling errors were introduced.

Beginning with the truncation of the KBR data, three different data sets were created with the KBR partials range (PR) extending to degree and order 120, 140 and 160 respectively. Since only the KBR partials were varied in this experiment, the GPS partials data were fixed at 40x40 for all cases. The square root degree variances (see Appendix D for details) of the resulting solutions, when compared to the truth field, can be seen in Figures 3.1 and 3.2. The degree variances (DV) show that the KBR truncation error is not a significant error source at or above degree and order 120. The degree difference variance (DDV) between the 120x120 case and the 160x160 case shows that the size of the truncation error is below that of the degree error variance (DEV), or formal errors. It is possible that there is some point at which this truncation error becomes more sizeable, but in terms of higher degree and order solutions, it is not a factor. The similarity of the three cases also suggests that any additional variations of the KBR partials within or above the range tested would have little effect on the resulting solutions.

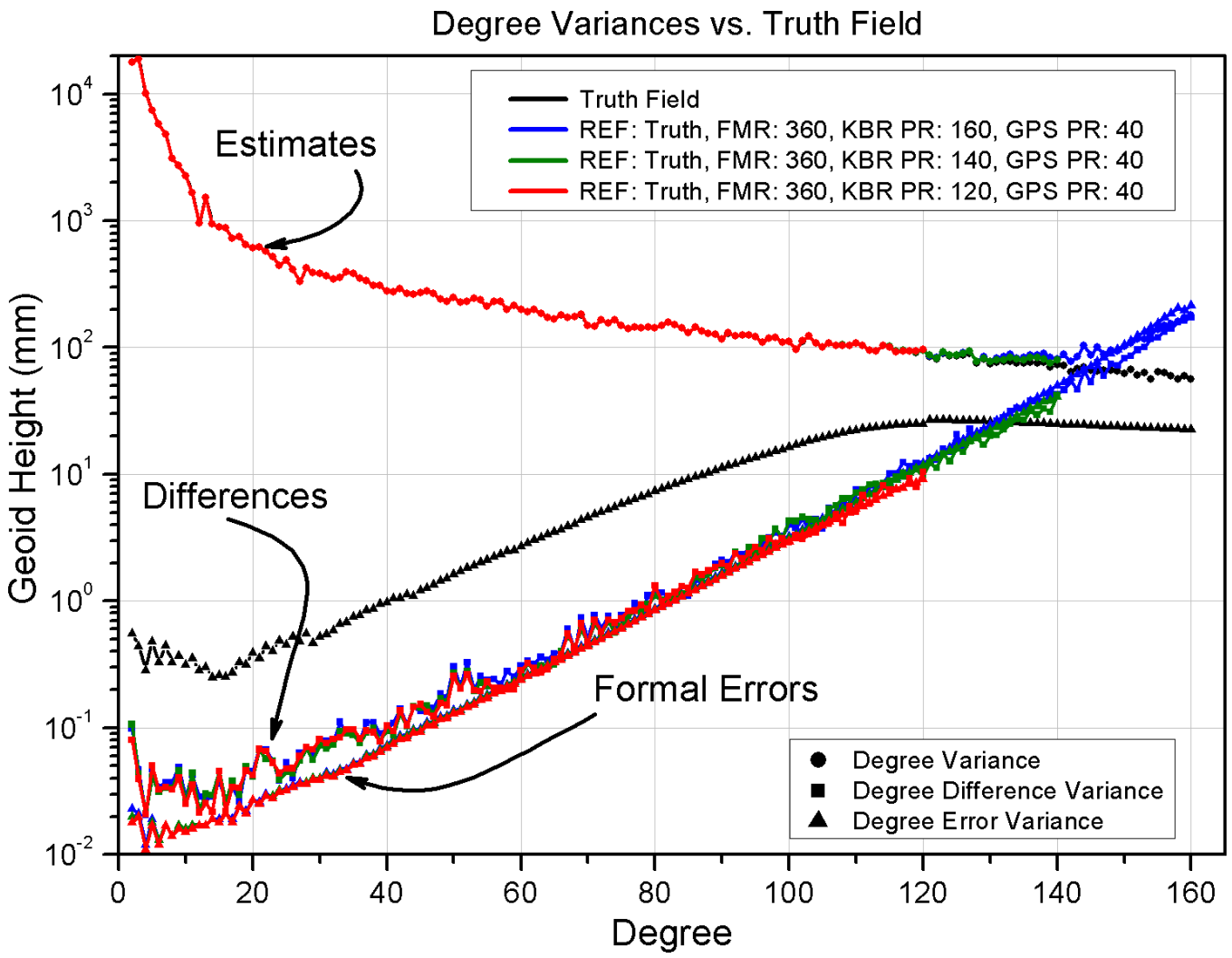


Figure 3.1: Simulation results, in terms of square root degree variances, for the case in which the KBR partials were truncated. For these experiments, no modeling errors were introduced.

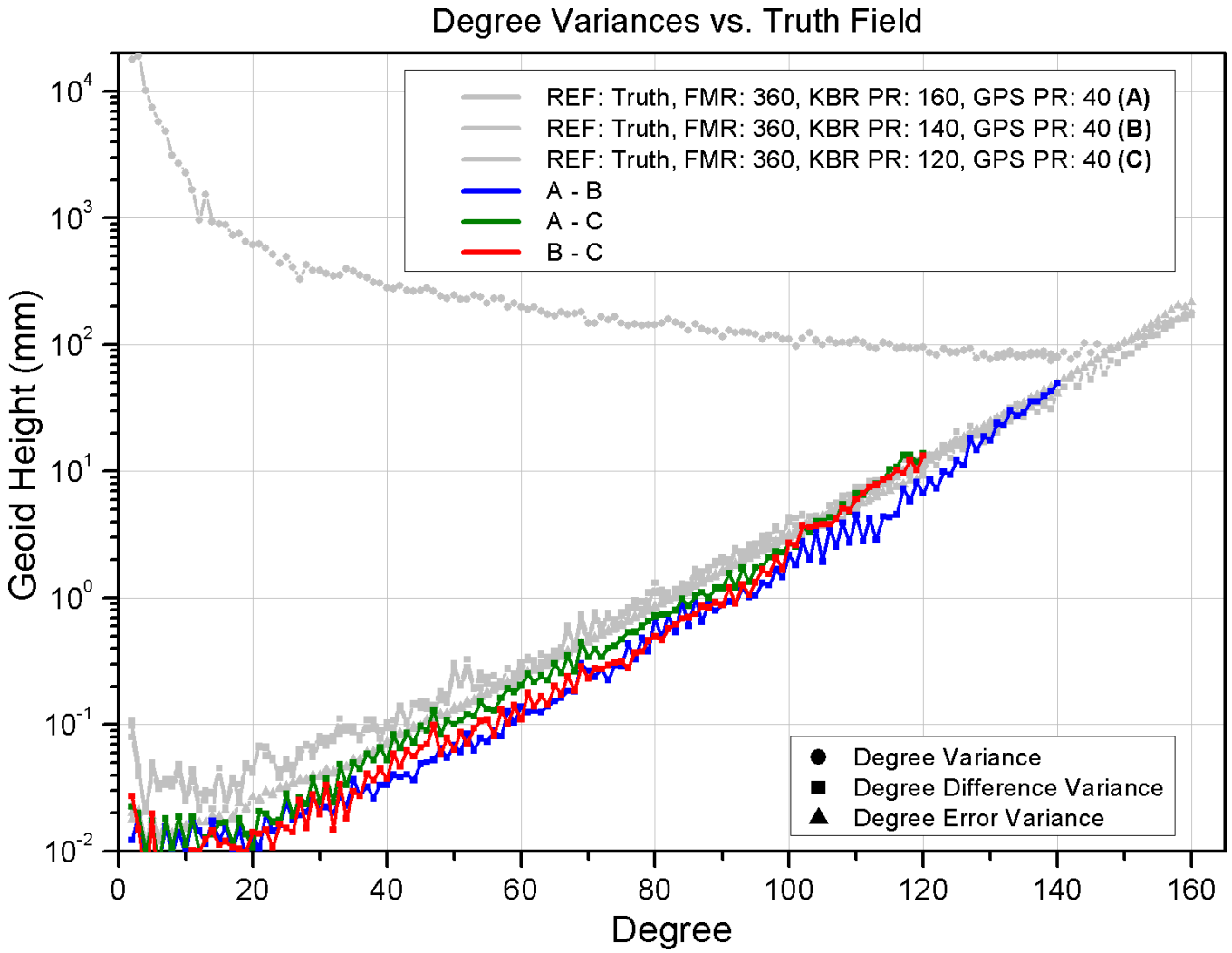


Figure 3.2: Degree difference variances of the truncated KBR fields. The differences fall below the formal errors, indicating that the truncation error is sufficiently small at the KBR degree bands evaluated.

Truncating the GPS data resulted in similar findings. Figure 3.3 compares the difference between the full 120x120 GPS partials case and the truncated 40x40 case. The KBR partials range was fixed at 120x120 for these experiments and, like the KBR truncation experiments, the truth field was used as nominal with a resolution of 360. As in 3.2, the DDV falls below the DEV of the full partials case, indicating that the errors in question are below the uncertainty in the solution.

While the truncation errors of the KBR and GPS in the presence of only measurement noise were not expected to be significant, the knowledge that they are close to negligible is helpful and makes the interpretation of the other simulation results easier. The absence of errors in the force model also allows these solutions to be treated as a baseline for the other simulations to be described later in this chapter.

It should be noted that while the truncation errors were inconsequential with only measurement noise applied, it will be shown later in Section 3.6 that the truncation of the GPS partials creates a noticeable artifact in the gravity field model when commission errors are introduced to the system. Additional studies, such as those described in Chapter 4, investigate this behavior using real GRACE data and offer techniques by which the commission error can be attenuated without having to increase the range of the GPS partials.

3.5 Omission Errors in the Force Model

To isolate the error of omission present in the nominal force model, another series of experiments were conducted in which the partials range for both the KBR and GPS data were fixed, but the force model resolution was allowed to

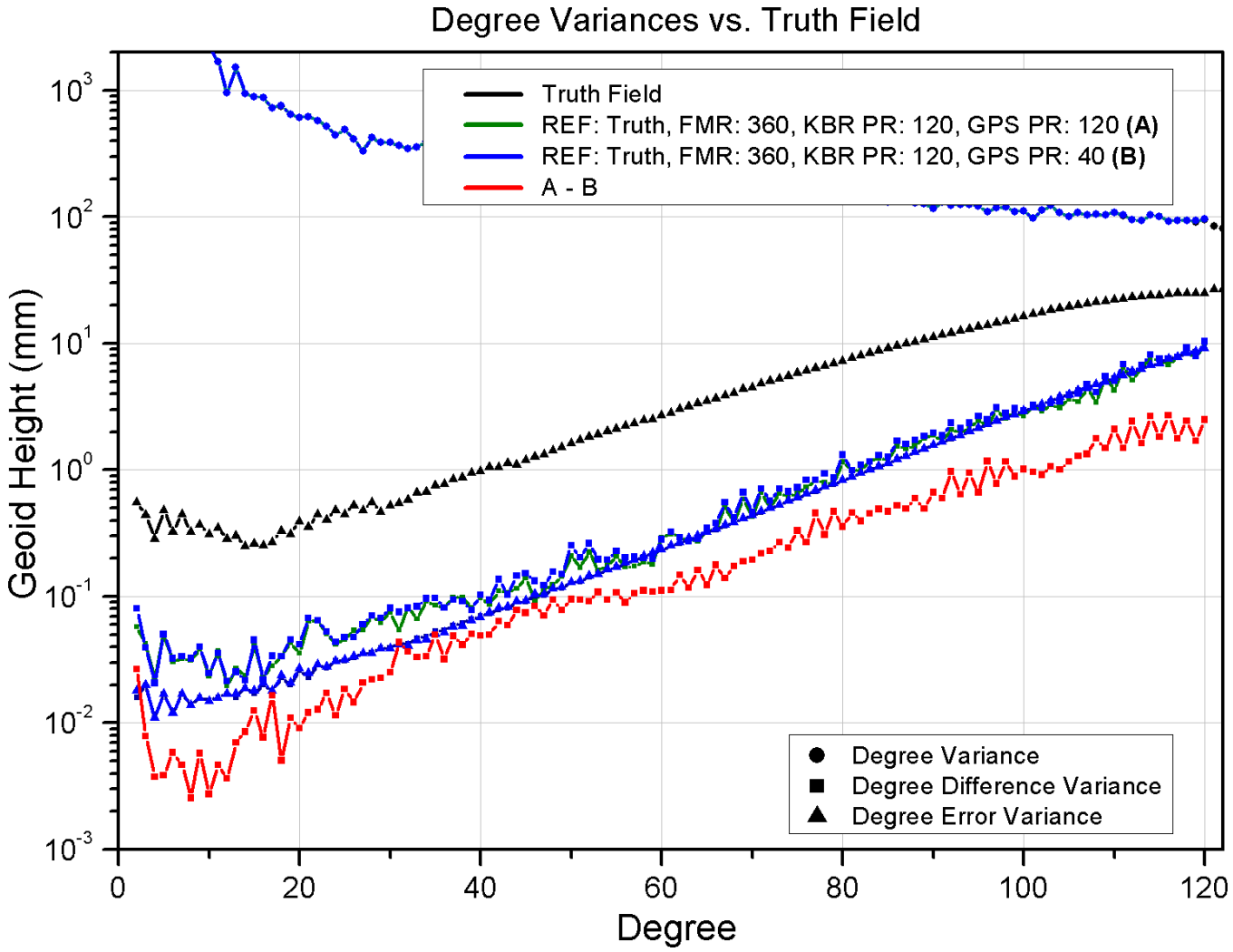


Figure 3.3: Simulation results for the case in which the GPS partials were truncated. No modeling errors were introduced. The difference between the full and truncated GPS partials cases are below the formal error of the solution.

change. Recall that the error of omission refers to the error associated with unmodeled forces. The truth field that created the observations used a force model out to 360x360, so to measure the omission error requires that a force model that is less than this be used. To avoid the introduction of commission errors, the truth field was used to generate three different data sets in which the size of the force model used to evaluate the partials was set to 120x120, 200x200, 280x280 and 360x360 respectively. The KBR partials were fixed at 120x120 and the GPS partials were fixed at 40x40. The results of the simulations can be seen in Figures 3.4 and 3.5. The solutions with the FMR set to 200, 280 and 360 were all very similar when compared to the truth field. The case in which the FMR was set to 120 shows evidence that the point at which the omission errors in the force model begin to become sizeable had been reached. The degree difference variance between the various cases is shown in Figure 3.5, and illustrates how the omission error grows as the FMR is reduced.

The simulation results imply that unmodeled forces beyond 200x200 do not significantly impact the gravity solution for the given set of measurement errors. This conclusion is limited by the fact that the truth model used for the simulations extends only out to 360x360. The Earth's gravity field has no such limitation; however, it will be shown in later chapters that these simulated results also compare closely with solutions done with actual GRACE data.

3.6 Commission Errors

The next experiment was designed to test the errors of commission, or those errors associated with *a priori* assumptions in the nominal model. The experiment involved running two cases, one using the truth reference field and the

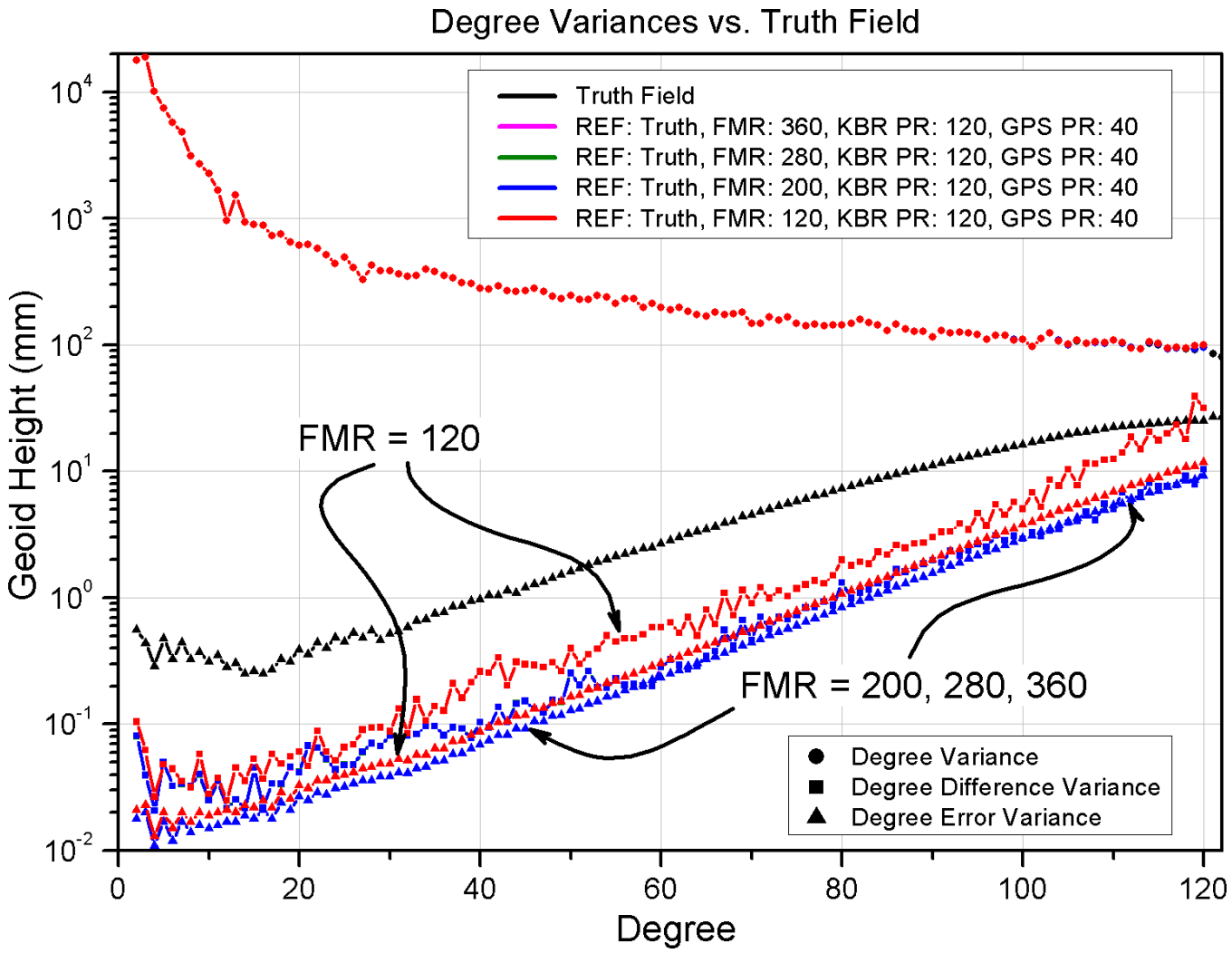


Figure 3.4: Degree variance plot showing the influence of changing the force model resolution for a fixed GPS and KBR parameter set. Only once the FMR was reduced to 120x120 were the omission errors noticeable.

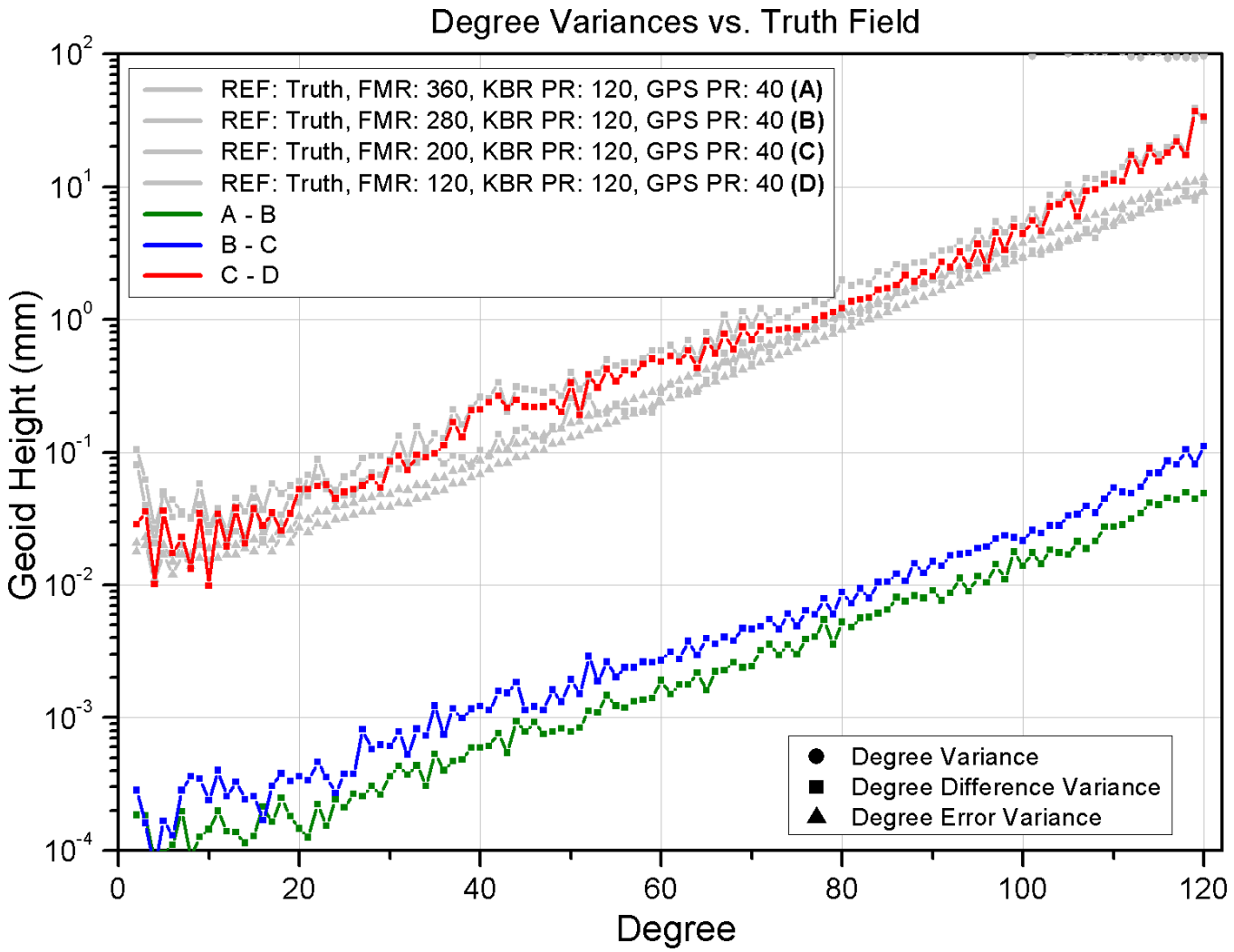


Figure 3.5: Plot illustrating the degree difference variance between the omission study solutions. The solutions with a FMR greater than or equal to 200 are very close to each other, indicating omission errors in the force model can be avoided by using a resolution above this level.

other using a clone reference field. Even though the previous omission studies indicated that the force model resolution (i.e., FMR) was sufficient at 200x200, the resolution for this experiment was maximized at 360x360 as a precaution. As before, the KBR partials were fixed at 120x120 and the GPS partials to 40x40. Figure 3.6 shows the solution results.

As indicated by the degree variances, the error of commission is not negligible. The most obvious features of the commission error can be seen by the appearance of two "bumps" around degrees 16 and 32. The fact that these "bumps" appear in solutions that used a FMR of 360x360 eliminates omission errors in the force model as a contributing factor. However, the fact that these features appear below degree 60 suggested that perhaps the GPS partials, which were limited to 40x40, were particularly sensitive to these errors. To explore these possibilities, another round of simulations were conducted in which the truncation of the GPS and KBR partials were once again tested, this time in the presence of commission error.

3.6.1 Combined Truncation and Commission Error

It was shown in Section 3.4 that the effect of truncation in the presence of only the measurement noise was not a significant error source. However, the appearance of the "bumps" in Figure 3.6 suggested that when the effects of modeling errors and truncation are combined, a noticeable artifact in the gravity model can be observed. To investigate this notion, another series of tests were conducted in which the truncation of the KBR and GPS partials were examined in the presence of commission error.

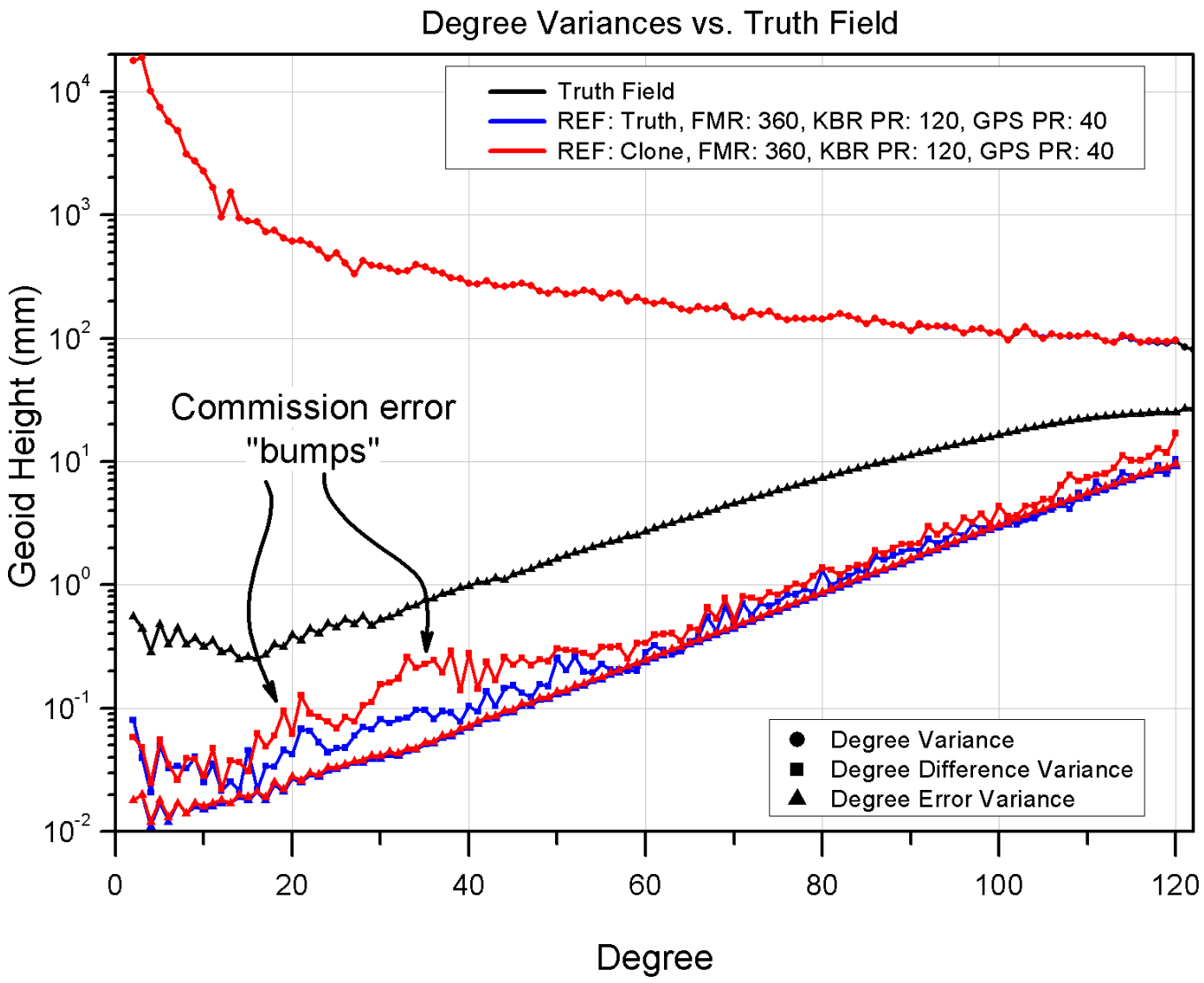


Figure 3.6: Degree variance plot illustrating the influence of commission errors. Note the presence of the commission error "bumps" at the low degrees.

Truncated KBR Partials with Commission Error

This first set of experiments truncated the KBR partials in the same sequence as in Section 3.4, i.e., at degree 120, 140 and 160. The force model resolution was fixed at 200x200 and the range of the GPS partials was set to 40x40. The clone nominal field was used to introduce commission error. The results of this experiment are shown in Figure 3.7. The three cases are all nearly identical in terms of degree variances, implying that the truncation of the KBR partials has little influence on the "bumps" observed at the lower degrees.

Truncated GPS Partials with Commission Error

Having examined the range of the KBR partials on the solution, the next step was to evaluate the effect of changing the range of the GPS partials. Since the original simulation had fixed the GPS partials at 40x40, the next logical step was to expand this range to something beyond this. Consequently, two additional simulations were run in which the GPS partials were extended to 70x70 and 120x120 respectively. For consistency with earlier solutions, the KBR partials were fixed at 120x120, the force model resolution was set to 200x200, and the clone field was used as the nominal field to introduce commission errors. The results are shown in Figure 3.8.

Extending the GPS partials had a noticeable effect on how the commission error was absorbed into the solution. It is interesting to see how the 70x70 GPS partials curve seems to relocate the commission error to a higher frequency than the 40x40 case. The 120x120 GPS partials case was clearly the best of the three cases, leaving no sizeable "bumps" in the degree difference curves.

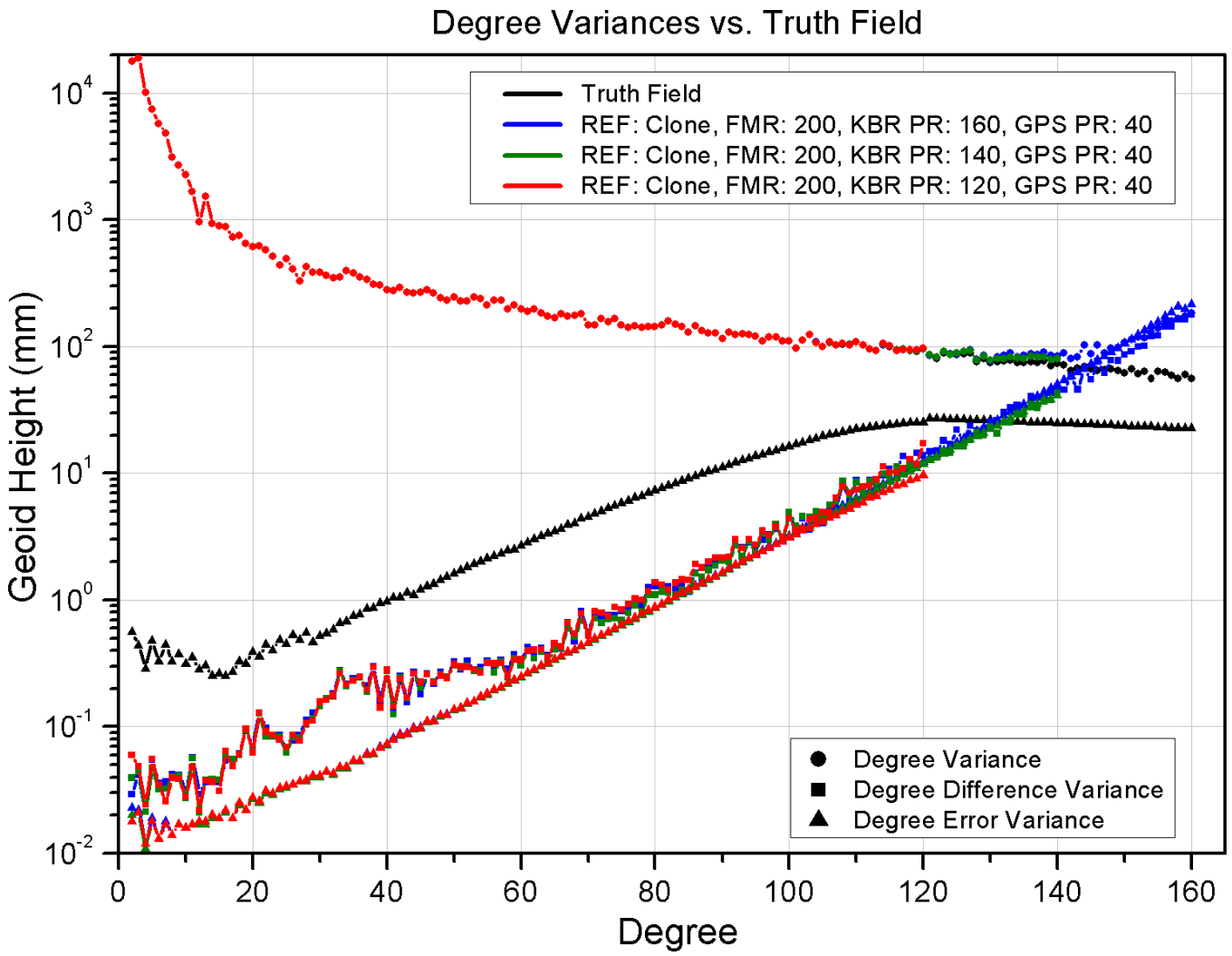


Figure 3.7: Degree variance plot showing the influence of truncating the KBR partials (with a fixed GPS parameter set) in the presence of commission error. No noticeable changes between the solutions are witnessed.

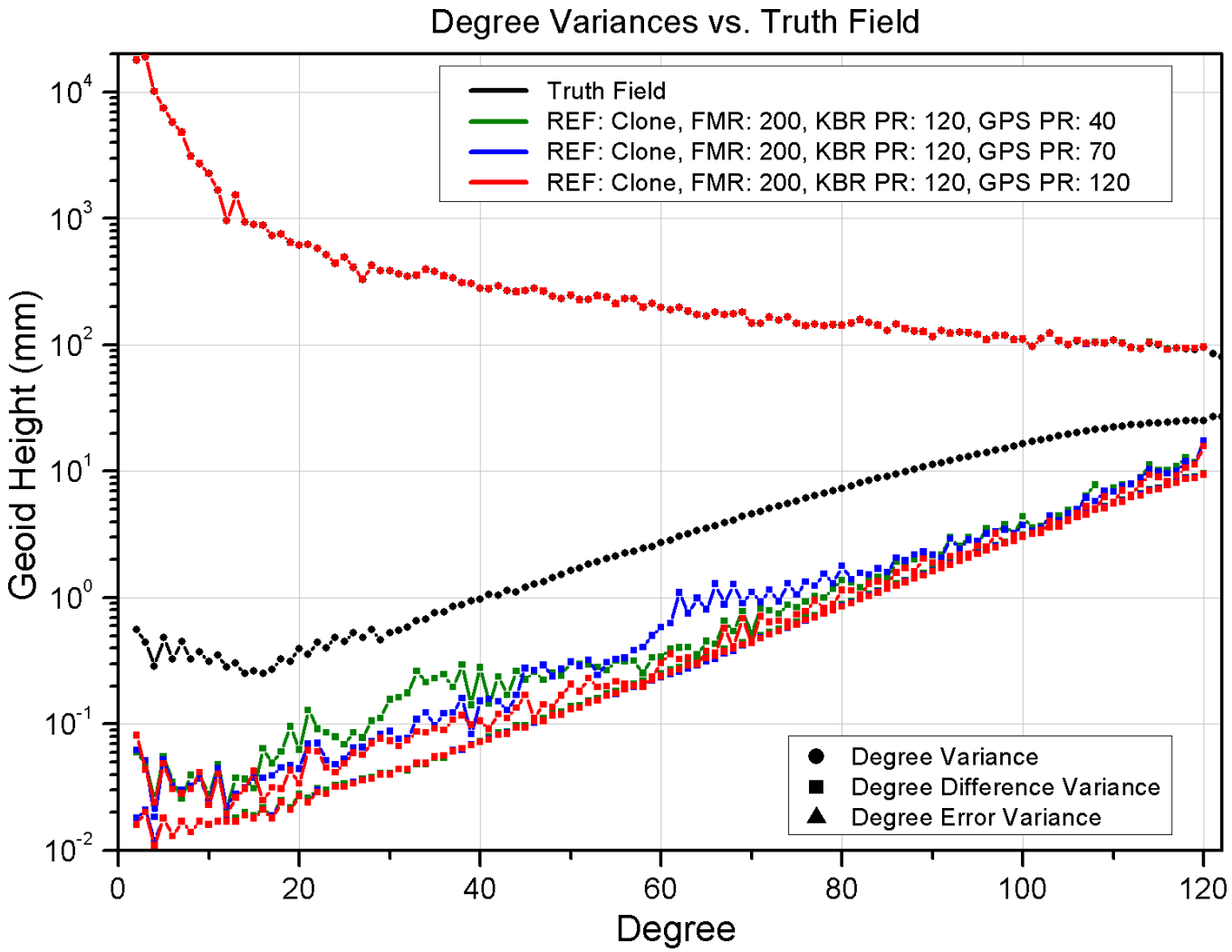


Figure 3.8: Degree variance plot showing the influence of truncating the GPS partials (with a fixed KBR parameter set) in the presence of commission error. Note how the "bumps" get shifted higher as the range of the GPS partials is increased, eventually disappearing when the range reaches its maximum of 120x120.

Figure 3.9 compares the 120x120 GPS partials case with the case in which a full GPS and KBR parameter set was used and no force model errors were introduced (i.e., the baseline). The important difference between the two curves is that one was created in the presence of commission error and the other was not. As can be seen in the plots, the case with commission error is very similar to the curve without commission error, with the difference falling below the formal errors, indicating that the higher GPS partials range was able to handle the effects of commission error much better than its lower range counterparts.

A set of gravity error maps, expressed in term of mm of geoid height, were created to better visualize the differences between the 120x120 and 40x40 GPS partials cases. Since all of the solutions involved were derived from simulated data, the maps have no real physical meaning, but can be useful in highlighting certain sensitivities not evident in the degree variance plots. These maps are shown in Figure 3.10. The top two maps are the two GPS partials cases differenced against the truth reference field, while the lower map shows the difference between these two cases.

The maps illustrate the sensitivity of the solutions to the commission error "bumps" described earlier. The smoother map for the 120x120 GPS partials case shows the benefit of extending the GPS partials range. These maps, as well as the degree variance plots, support the conclusion that the full 120x120 GPS partials case was able to accommodate the commission error much better than the truncated GPS cases.

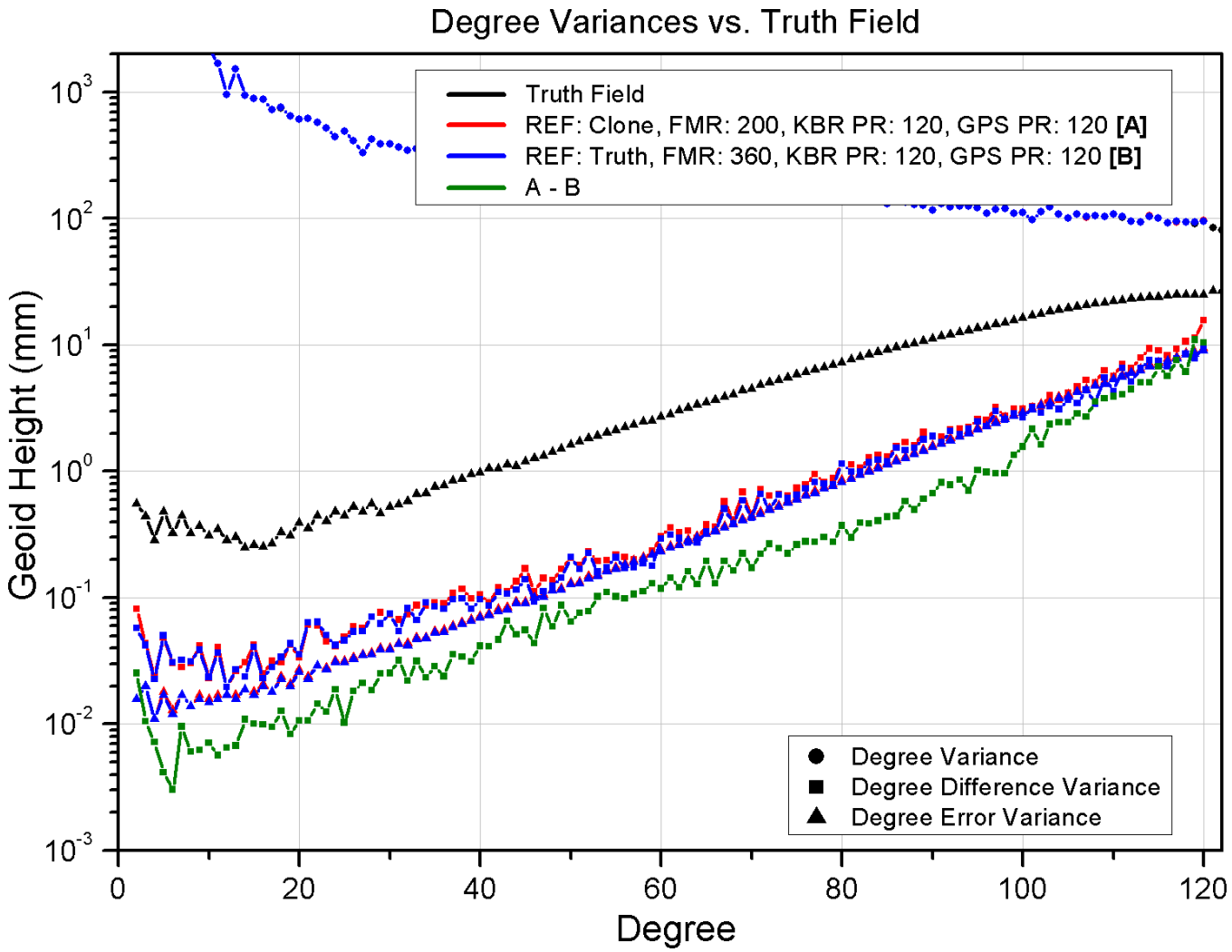


Figure 3.9: Degree variance plot showing the benefit of extending the GPS partials. The 120x120 GPS partials case, created in the presence of commission error, removed the "bumps" observed in earlier simulations to the point that the solution is nearly identical to the case in which no commission error was used.

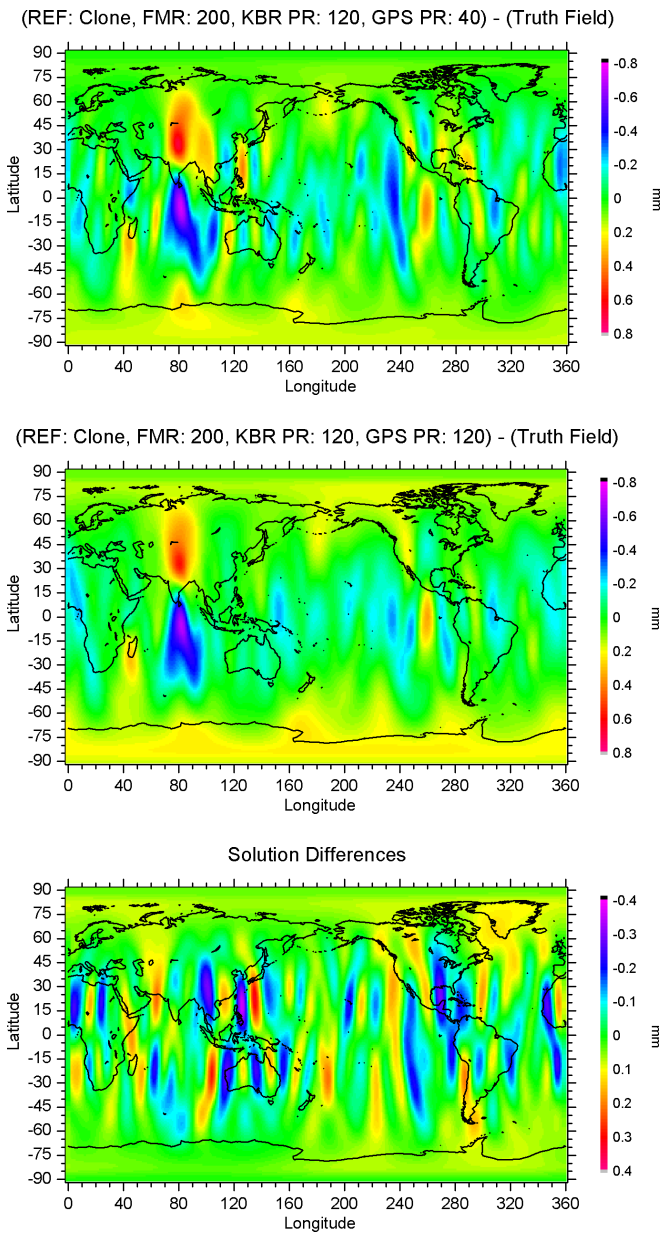


Figure 3.10: Gravity error, expressed in terms of mm of geoid height, for the 120x120 and 40x40 GPS partials cases. A 600 km radius smoothing was applied. The top two panels highlight the error with respect to the truth field. The lower plot (note the scale change) shows the difference between the two cases.

3.7 Conclusion

For this chapter, a number of simulations were run to assess the impact of certain types of estimation errors, namely the errors of omission and commission. Through the first set of experiments, it was discovered that the omission error due to the discretization of the geopotential model, or the truncation error, is not a significant error source when isolated, i.e., in the absence of errors in the force model. While not completely negligible, the isolated truncation errors fall below the formal errors and are considered small enough to be of little concern.

Another series of experiments examined the influence of another form of omission error related to unmodeled forces in the nominal field. When the force model resolution is relatively low, i.e., 120x120 or below, the omission error is large enough that it can have a noticeable impact on the solution. If the force model resolution is above 200x200, which is the case for the typical GRACE RL01 processing scenario, the omission error is small enough to be considered negligible.

An analysis into the errors of commission, or errors due to imperfect assumptions in the force models, showed that the GPS data is particularly sensitive to this error source. Using a reduced GPS parameterization of 40x40 and 70x70 showed a measurable influence from the commission error. It was only by using an extended GPS parameterization (i.e., 120x120 for these particular simulations) that the commission error was sufficiently attenuated; however, this is not the only means by which this can be accomplished. As will be seen in the next chapter, the same result can be achieved by a technique in which the GPS data is downweighted appropriately.

As a final note, it is important to remember that given a more accurate nominal model, the commission errors would not be as large, and might not require the extended GPS partials range. As the GRACE mission continues to collect data and refine its models, the nominal fields will improve and the effect of commission errors will most likely be reduced. To verify this notion, another set of simulations were run in which the size of the commission errors were reduced by 50% and 75% from its original level. To emulate the RL01 processing environment, the simulations contained errors of commission, omission and truncation. Figure 3.11 show the degree difference variances when the reduced commission solutions are compared to the no-error baseline case of Section 3.3. As the level of commission error decreases, the impact of the reduced GPS partials range becomes smaller, to the point that it is below the formal errors of the solution in the 75% reduction case. Similar results were achieved by Kim in his studies [37, p. 225].

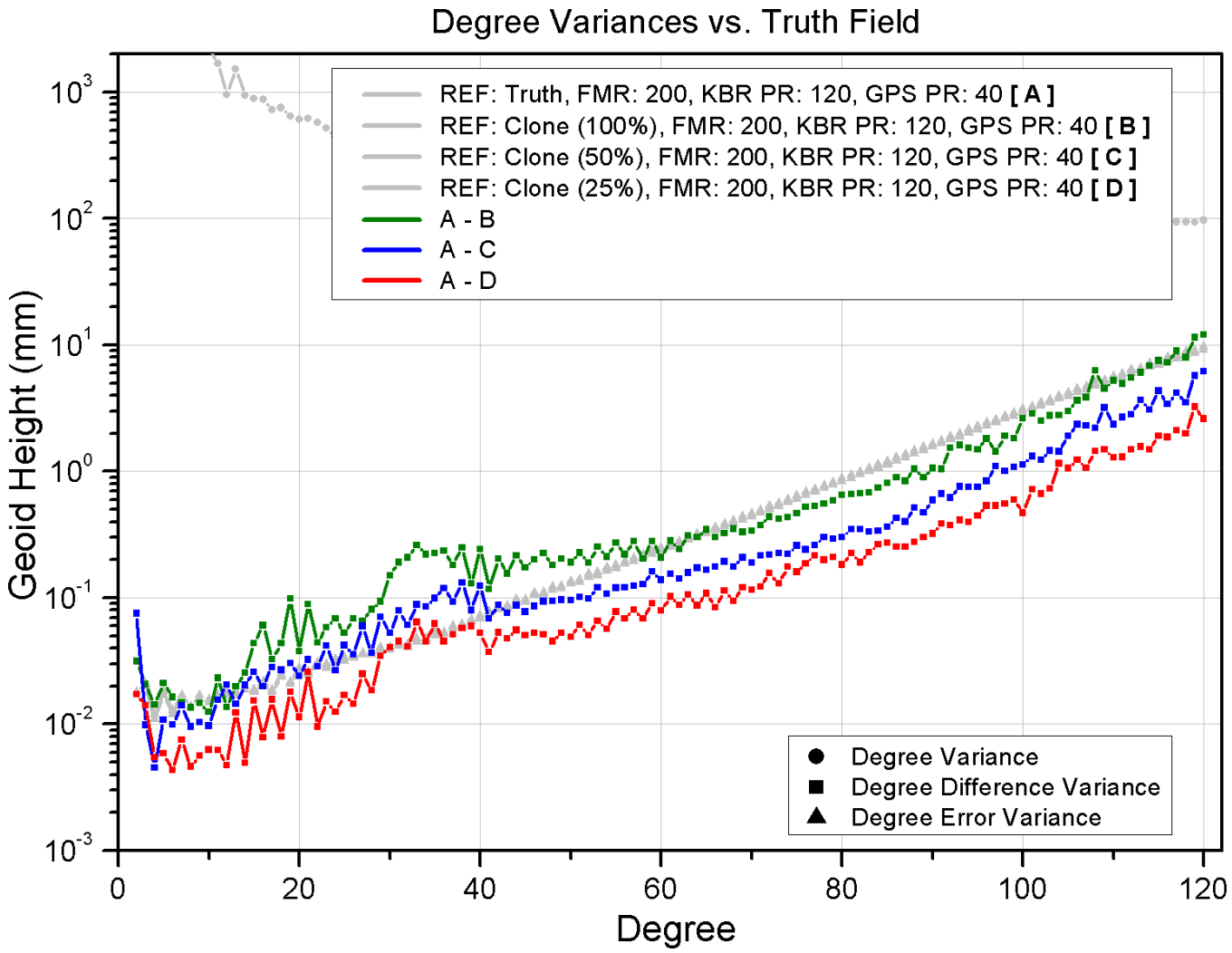


Figure 3.11: Degree variance plot showing the benefit of a more accurate nominal model. As the error in the nominal field approaches that of the truth, the impact of the truncated GPS partials is reduced.

Chapter 4

The Treatment of GPS Data

4.1 Introduction

In this chapter, the combination of the GPS data with the K-band range (KBR) data was examined in the context of the GRACE first release (RL01) processing scenario[1]. This analysis was motivated initially by the need to reduce the size of the GPS data files involved in the estimation process. The first GRACE gravity field models made use of all available measurement data and allowed both the GPS and KBR data sets to estimate the full range of gravity coefficients. Using this approach, the majority of the computational resources (both cycles and disk storage) were being devoted to processing the large volume of GPS data. The high orbits of the GPS satellites limit the sensitivity of the double-differenced observations to high frequency gravity perturbations, making it inefficient to spend the bulk of the processing effort on the GPS measurements. As a result, various techniques were explored in an attempt to reduce the scope and number of the GPS measurements while still retaining their value in the gravity recovery process. These included a random decimation of the GPS observations, reducing the size of the GPS ground station network, and truncating the partials range of the GPS data.

[1]Data processing standards and a user handbook for the GRACE gravity solutions are available at `www.csr.utexas.edu/grace/publications/handbook`

The simulations of Chapter 3 showed that truncating the GPS partials in the presence of commission error had a negative impact on the resulting gravity field model. This same behavior would also be observed when processing real GRACE (RL01) data, resulting in the development of a processing strategy in which the GPS is both truncated and artificially downweighted. The benefit of this approach was verified through the analysis of real and simulated data.

The results of these experiments will demonstrate how a noticeable *improvement* in the gravity field can be achieved while also decreasing the total processing time for an average GRACE solution by roughly 75%.

4.2 Experiment Details

Unlike the previous chapter, the focus of this chapter will be on the treatment of real GRACE data. The experiments described in this chapter were conducted using GRACE RL01 data collected in either August 2002 (22 days) or April 2003 (26 days). Each data set consisted of two types of measurements: inter-satellite K-band range-rate measurements (5 sec sampling rate) and GPS double-differenced (GPSDD) measurements (30 sec sampling rate) collected from a 43 station ground network. A brief description of the RL01 gravity estimation process is provided below.

It is first assumed that the raw instrument measurement data from the GRACE satellites have been collected and processed such that there exists two sets of measurements partials, one for the KBR and GPS data. A one day batch estimation period was used to create these measurement partials files. To improve the estimate of the gravity field, the GPS orbits are first converged with a reference gravity field and a full network of GPS ground stations, then

the gravity parameters are allowed to adjust during the least squares accumulation phase [52]. Both the KBR and GPS data were edited to remove outliers and other instrument errors. A standard parameterization was used for the GPS data, including phase ambiguity, zenith delay parameters, and orbit element corrections (OECs). For the KBR data, the long wavelength errors were accommodated by a set of bias, slope and once-per-revolution parameters [37]. Common to both data types was a set of state parameters (i.e., position, velocity), as well as accelerometer scale factors and biases. During the least squares reduction phase, the data sets for each day were then optimally weighted [72] based on their computed post-fit residuals (iterating until converging to a given criteria). Table 4.1 provides a summary of the parameterization used for these experiments, including the type and duration over which the parameters were estimated. For reference, MSODP [46] version 2002.1 was used for the orbit convergence and partials generation phase and AESoP 1.4.x was used to perform the least squares estimate of the models. The KBR measurement partials were extended to degree and order 120, with the GPS partials taken out to degree and order 70 or less. The solutions were done using only GRACE RL01 data and without the use of *a priori* information for the gravity coefficients.

4.3 Evaluating the Gravity Solution

In a simulated environment, the impact of a change to the system can be easily found because the true answer is known. When working with real data, the truth is not known and any improvement or degradation in the final result must be determined through a much more subjective process. For the case of evaluating gravity field solutions, this is accomplished by observing the per-

Parameter Type	Abbreviation	Duration
Accelerometer bias	AC0	1 day
Accelerometer scale	AC1	30 days
GPS DD Ambiguities	DD AMB	Per cycle slip
GPS Zenith Delay	DD ZEN	As needed
Low-low bias	LLB	45 min
Low-low bias periodic	LLBP	90 min
Low-low bias rate	LLBD	45 min
Initial conditions	IC	1 day
Gravity coefficients	GEO	30 days

Table 4.1: Parameterization used in the real GRACE data experiments of this chapter.

formance of a candidate field when subjected to a range of independent tests. The three primary tests used to evaluate the results of this chapter include the square root degree variance, orbit fit test and ocean circulation test. A full description of these tests can be found in Appendix D. The square root degree variance, seen earlier in Chapter 3, computes the sum of squares of the estimates or uncertainties by degree for a given solution. It is a useful statistic for showing power as a function of degree. The orbit fit test uses the candidate field to estimate the orbit of a satellite in conjunction with available satellite laser ranging (SLR) observations. A root mean squared (RMS) value is then generated, with a low RMS value indicating a better orbit fit for the given satellite. The better the gravity model, the better the fit. Lastly, the ocean circulation test compares the dynamic ocean topography (DOT) map computed from the candidate field to the DOT computed from in situ ocean data. The results of this tests are expressed in terms of an RMS and correlations. All of these tests have limitations and are only used to evaluate specific qualities of a field. For example, the orbit tests are most valuable for evaluating the

performance of a field at the low degrees, while the ocean circulation tests are useful in evaluating the mid-degree performance of a field. By using this suite of tests, a general assessment regarding the quality of a field can usually be made.

4.4 Random Decimation of the GPS Observations

It was known well before its launch in March, 2002, that the GRACE mission would generate a large amount of data from both the High Accuracy Inter-satellite Ranging System (HAIRS) as well as from the onboard GPS receivers and corresponding ground tracking stations. If the full network of 43 GPS ground stations are utilized, it is possible to generate upwards of 150,000 GPS double-differenced measurements per day. In addition to these, the inter-satellite ranging measurements are filtered and sampled at 5 second intervals, creating an additional 17,000 observations per day. To create a gravity field model from these observations requires the numerical integration of hundreds of thousands of differential equations as well as the dense least squares estimation of the resulting linearized measurement partials (see Section A.2). The least squares accumulation process alone is an $O(n^3)$ operation and the disproportionately large number of GPS observations was initially consuming roughly 90% of the total solution compute time. This seemed unnecessary considering that the GPS data are not the primary observable of the GRACE mission and are needed primarily to aid only in the determination of the long wavelength signals and satellite positioning.

As a result, one of the goals of this study was to determine if the number of GPS observations could be reduced without affecting the quality of the

gravity field solution. The most direct way of reducing the number of GPS observations involved in the solution process is to simply remove them. Preliminary simulations showed that the number of GPSDD observations could be dramatically reduced without substantially impacting the gravity solution. Based on these, a set of real data experiments were conducted on the April data set in which the GPSDD observations were randomly deleted from the estimation process. This was accomplished by arbitrarily skipping records as they were listed in the measurement partials files. Figure 4.4 shows how the arbitrary skipping of GPSDD observations (i.e., using only one of every fourth point, one every eight points, etc.) in the April data set had very little impact on the degree variances. The point of these experiments was mainly to investigate how the solution would react to a significantly lower number of GPSDD observations.

4.5 Reduced GPS Ground Station Network

The results of the previous section showed that the number of observations could be substantially reduced without dramatically impacting the solution. Another more systematic method of decreasing the number of GPSDD observations is to reduce the number of ground stations involved in the double difference procedure. Based on this, an initial reduced network of twelve stations was chosen from the full network of stations. Since the reduced network is roughly one quarter the size of the full network, it was important that each station produce a sufficient number of high quality observations. Most stations have well determined coordinates and collect accurate measurements, so reliability was not an issue, but the number of observations collected by each station

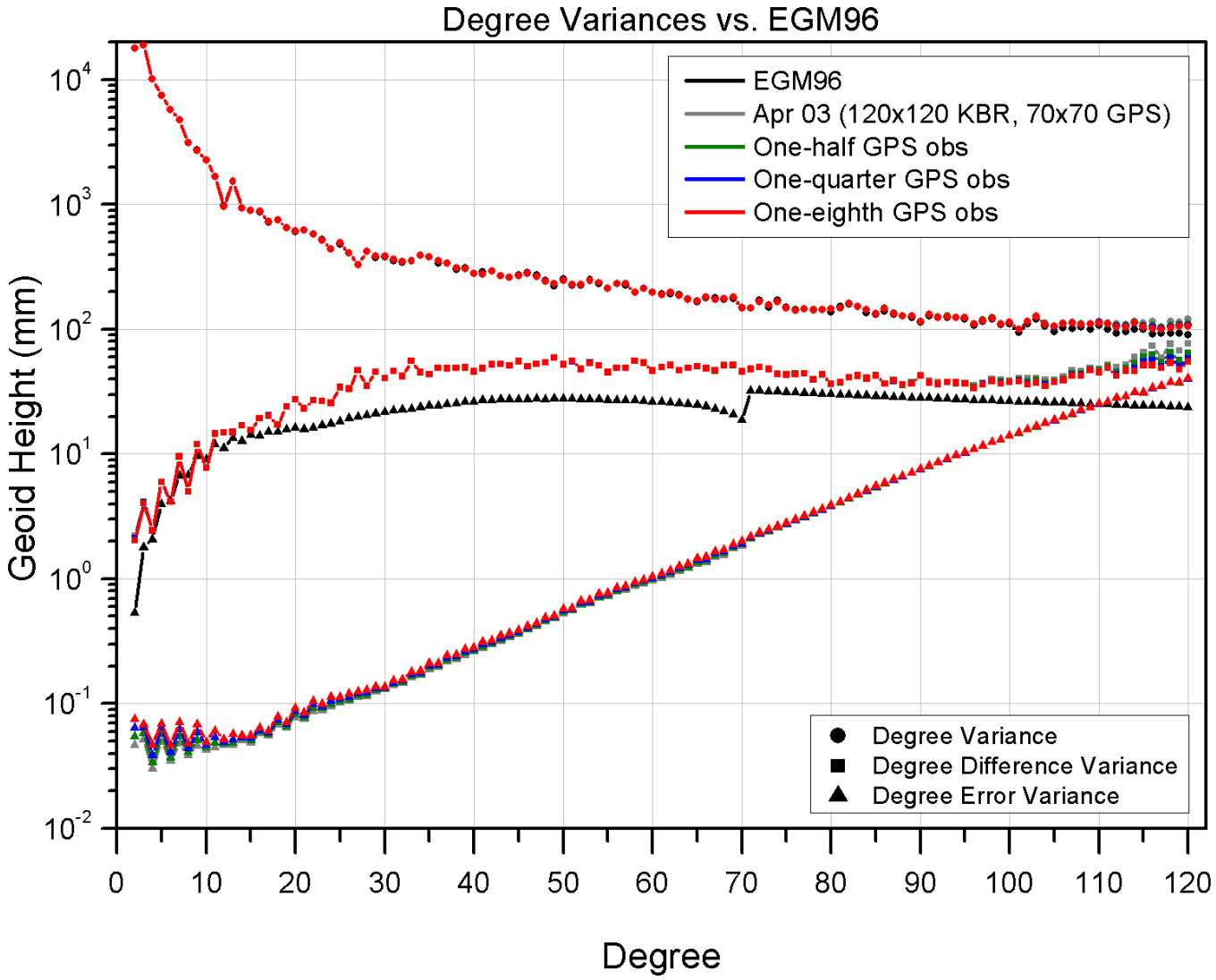


Figure 4.1: Comparison of solutions in which the GPSDD data for April, 2003, was decimated by the arbitrary use of every second, fourth, or eighth observation.

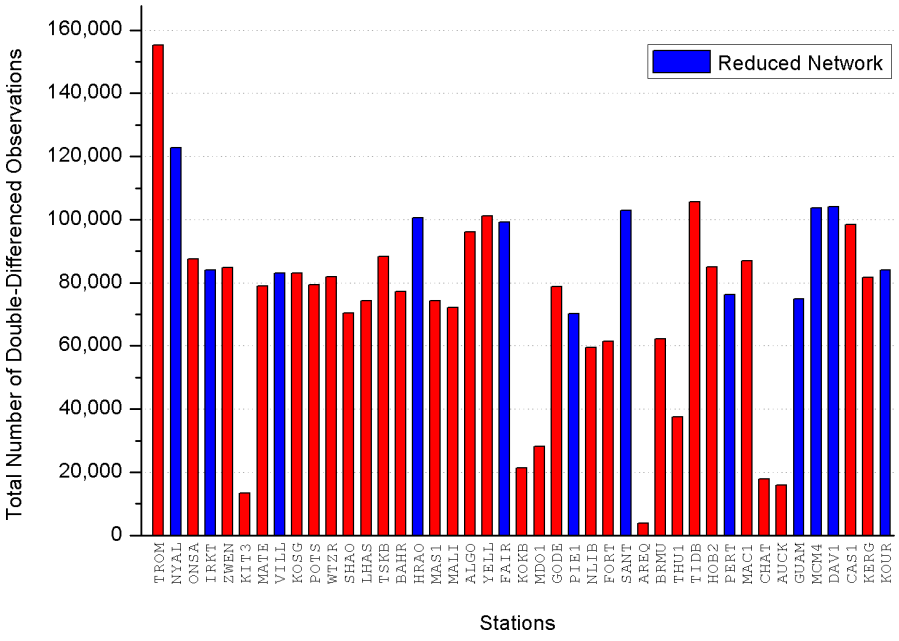


Figure 4.2: Number of GPSDD observations collected over the 22 day August test case for each station

did vary. Figure 4.2 shows the total GPSDD observation count for the 22 day August test case for each station. Only those stations which produced a sufficient amount of observations were admitted into the reduced station network. Geographic location was the other major criteria for choosing the reduced network. With fewer stations, it is important that the stations that are chosen provide sufficient coverage to adequately resolve the low degree harmonics. A set of randomly selected GPS stations could potentially weaken or bias the gravity solution, so the stations chosen for the reduced network had to have a good geographic distribution. There is also the possibility that one or more of the chosen stations could temporarily go out of service, potentially affecting the surface coverage. With this in mind, a series of experiments were done with a reduced network of twelve, nine and six stations. Figure 4.3 shows the

geographic locations of the full network as well as the various sub-networks. This figure also shows the GPS visibility mask for each station with a 15 degree elevation criteria. A gravity field solution to degree and order 120 was created using the August data for each of the reduced networks, the results of which are illustrated in Figure 4.4. In all cases, the KBR data remained the same with only the number of GPSDD observations changing. In addition, the GPS partials were only taken out to degree and order 70 for this particular experiment. For comparison, the solutions were differenced against the EGM96 gravity model [40]. The changes in the degree two terms as well as at the high degrees suggest that the use of a reduced network produces a slightly improved gravity solution over the full network case.

For the sake of robustness, the twelve station network is preferred, but a six station network should be more than sufficient to resolve the long wavelength signals. That is, of course, as long as the stations involved continue to produce a sufficient number of quality measurements. The twelve station network reduces the total number of GPS observations from 150,000 per day down to roughly 50,000 observations, a savings of 67%.

4.6 Reduced GPS Parameterization

In addition to decreasing the number of GPS observations through the use of a reduced network, a series of experiments were conducted to test the effect of the GPS parameterization on the solution. A smaller GPS parameterization would require the solution of significantly fewer parameters, decreasing the processing time of the GPS data. Decreasing the range of the GPS partials would also test the influence of the GPS measurements on the mid to high degree terms.

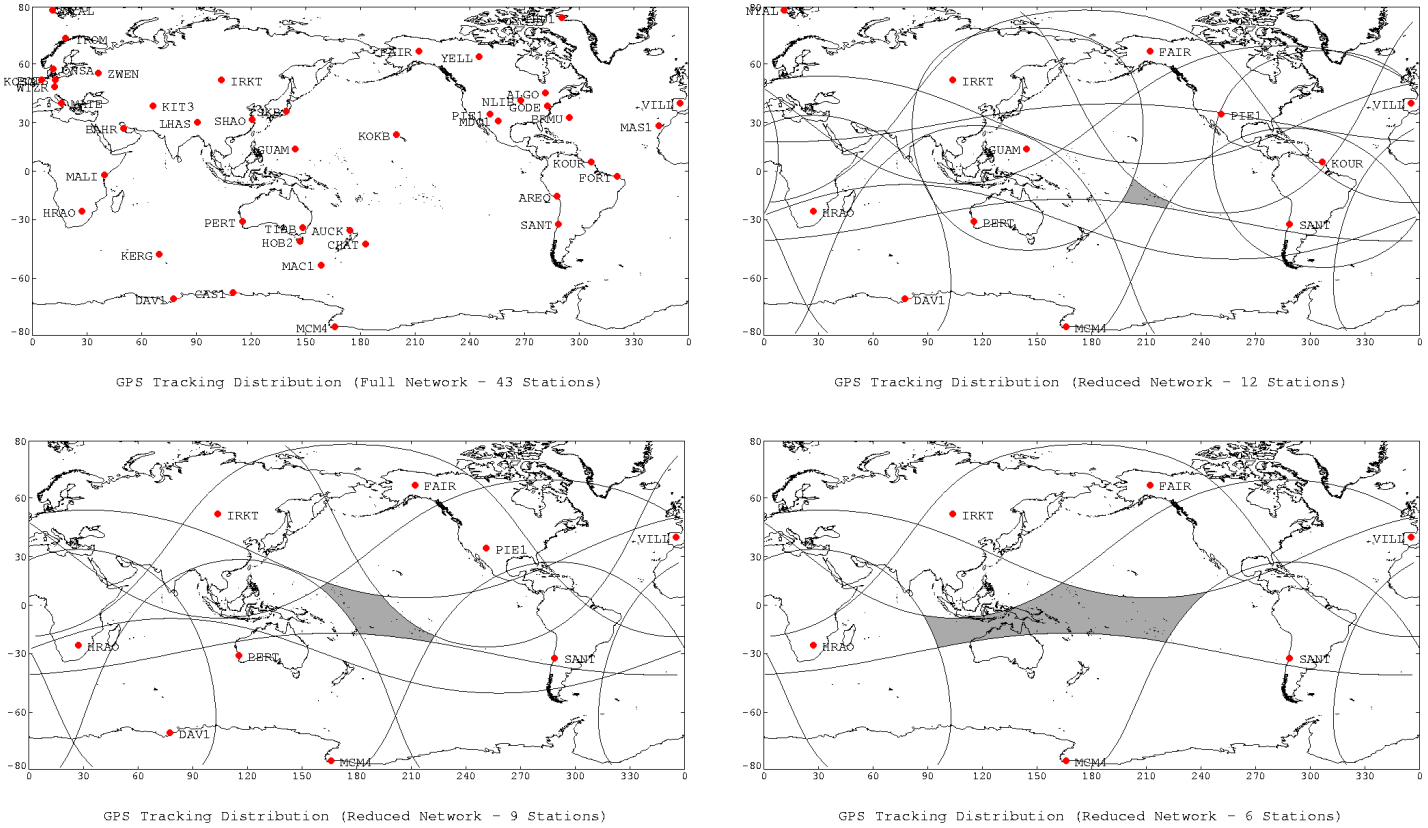


Figure 4.3: GPS station networks along with GPS visibility mask (15 degree elevation criteria). Shaded areas represent surface coverage gaps.

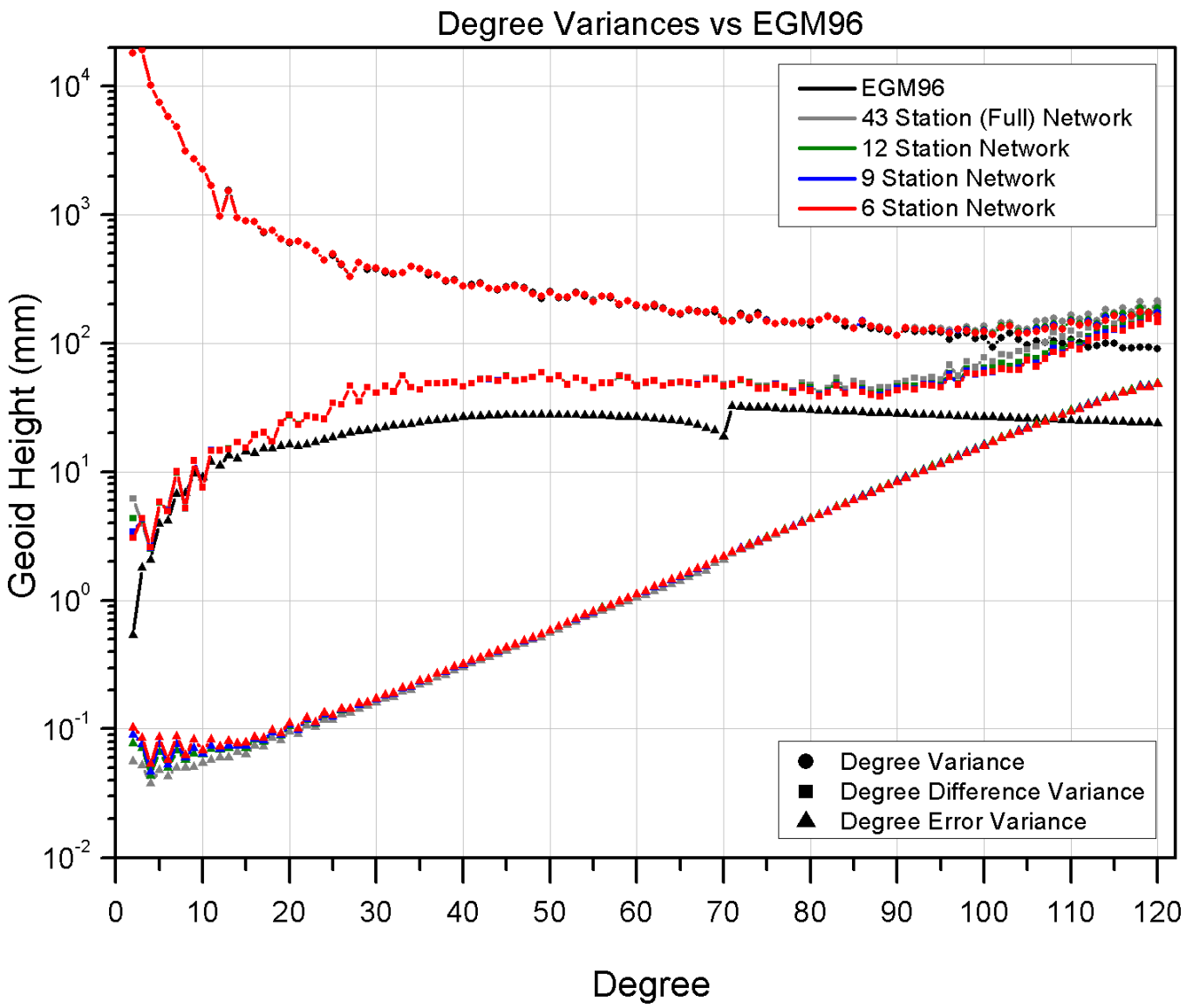


Figure 4.4: August, 2002, solutions utilizing various GPS ground station networks.

Due to their high altitude, the GPS measurements should not be sensitive to high degree and order variations. This is illustrated in Figure 4.5, which shows a solution done using only the August GPS data. The estimates and formal errors begin to degrade significantly at roughly degree 35, indicating that the GPS data should have no useful information beyond degree 40 or 50. Also included in this figure is a plot of one year's worth of GPS data taken from the CHAMP mission [6]. The GRACE satellites have nearly identical on-board GPS receivers as CHAMP, so Figure 4.5 suggests that even with a long term GPS data set, the power of the GPS measurements is limited to well below degree 70.

To verify this, a set of solutions were generated using the April, 2003, data in which the partials range of the GPS data was decreased. As before, the KBR data were not altered, but the GPS and KBR data were optimally weighted. The GPS data were created with the twelve station network discussed in the previous section. The results of this test are presented in Figure 4.6. The formal error variances for the 10x10 and 20x20 cases suffer significantly between degrees 15 and 20, possibly the result of not including the first and second resonant orders. The formal errors of the 40x40 case, however, are nearly identical to the 70x70 and 120x120 cases. The orbit and ocean circulation statistics for these cases are shown in Tables 4.2 and 4.3. For the most part, the 40x40 and 120x120 orbit test results are roughly equivalent. Interestingly, the 70x70 case performed slightly worse than all of the other cases, although not by a significant amount. The ocean circulation test favored the 120x120 case, with the 40x40 case performing worst.

It is interesting to note that the determination of J2 seems to improve

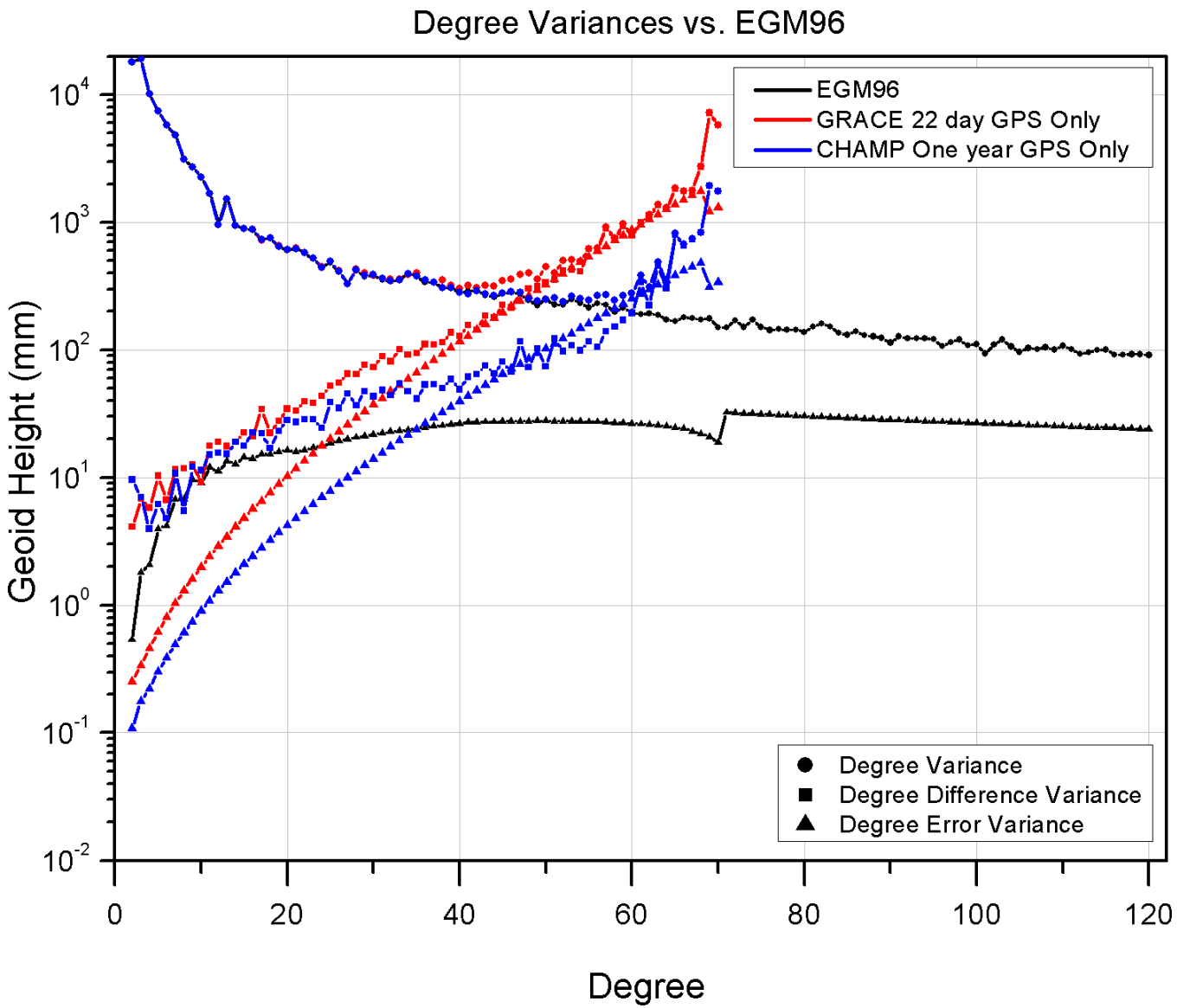


Figure 4.5: GPS only solution from the 22 day, August, 2002, GRACE data and a one year CHAMP solution.

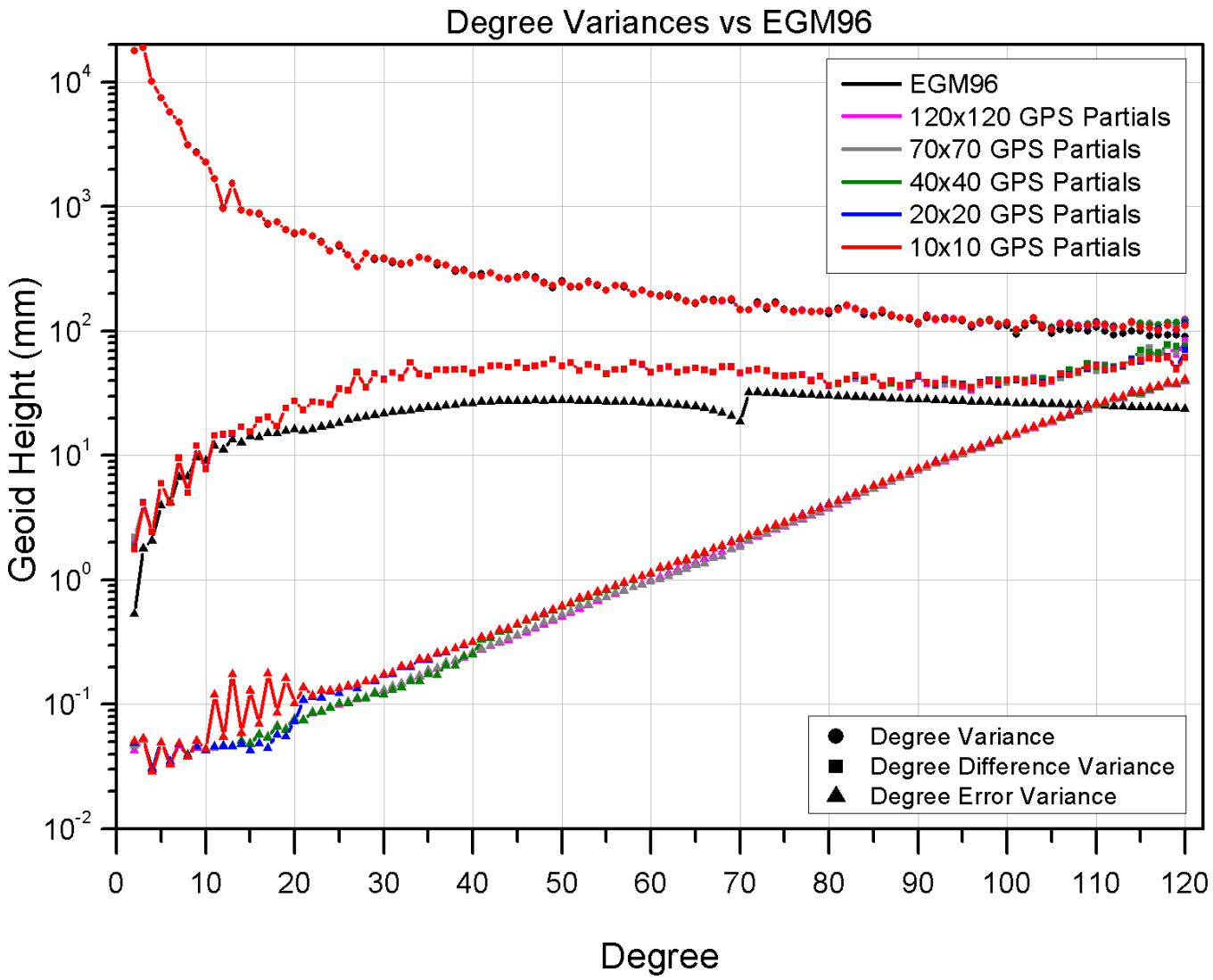


Figure 4.6: Twelve station reduced network solutions with varying GPS parameterizations. The solutions were generated from the 26 day, April, 2003, data set.

Satellite	1/revs	120x120	70x70	40x40	20x20	10x10
GEOS-3		6.8	6.8	6.9	6.9	6.8
GFZ-1		34.6	34.7	34.5	34.4	34.4
GFZ-1	yes	7.8	7.8	7.8	7.8	7.8
Lageos 1		1.09	1.14	1.09	1.06	1.06
Lageos 2		1.03	1.1	1.07	1.04	0.97
Starlette		8.4	9.6	8.5	8.4	8.3
Starlette	yes	2.8	2.8	2.8	2.8	2.7
Stella		8.7	9.4	8.5	8.4	8.3
Stella	yes	3.0	3.0	2.8	3.4	3.1
Westpac		7.9	8.2	7.8	6.9	7.7
Westpac	yes	5.1	5.2	5.3	4.5	5.2

Table 4.2: Orbit test results for the April reduced GPS partials experiments using a select group of satellites. Table numbers represent RMS values in units of cm.

slightly with the reduced partials cases. This is based primarily on the results of the Lageos orbit test results. The Lageos satellites are heavy, cannonball-style satellites that are particularly sensitive to the J2 perturbations. The orbit fits for the 10x10 reduced partials case show a roughly 5% improvement in the Lageos fits over the full 120x120 case. The Lageos I fits for the 20x20 and 40x40 are also equal to or better than the 120x120 case.

While the effect of the resonances would certainly need to be explored more in-depth, these tests paint a somewhat mixed picture with regards to the use of a reduced parameterization for the GPS data. The formal errors of Figure 4.6 indicate that the 40x40 and higher reduced partials cases fit the data to the same level as the 120x120 case. The orbit fits for the 40x40 case are on par with the 120x120 case, implying that the recovery of the low degree signals is not being hampered by reducing the GPS partials. On the other hand, the lower correlations on ocean circulation tests indicate that the reduced partials

Case	Zonal RMS (cm/sec)	Zonal Correlation Correlation	Merid RMS (cm/sec)	Meridional Correlation
120x120	2.55	0.93	3.04	0.510
70x70	2.59	0.93	3.13	0.487
40x40	2.57	0.93	3.18	0.476
20x20	2.58	0.93	3.15	0.488
10x10	2.59	0.93	3.16	0.485

Table 4.3: Ocean circulation statistics for the April reduced GPS partials experiments.

cases performed slightly worse than the full 120x120 case. One explanation for this might lie with the results of Section 3.6.1. These simulations showed that the GPS data were particularly sensitive to errors of commission, and that reducing the GPS partials range magnified the effect of these errors. To find out if this might also be affecting the solutions for these real data solutions, the degree difference variances between the 40x40, 70x70 and 120x120 reduced partials cases were computed, as shown in Figure 4.7.

If the 120x120 case is assumed to be more accurate, which is not unreasonable considering that this case tested equivalently in the orbit test and best in the ocean circulation tests, then this figure is useful in interpreting some of the results seen earlier. For example, the large difference at degree 2 for the 70x70 case explains why the orbit tests for the 70x70 case were poor. The differences in the 40x40 and 70x70 cases at the mid-degrees also explains why the ocean circulation tests were different between the 120x120 and other reduced partials cases. By increasing the GPS partials range from 40x40 to 70x70, we see that the differences below about degree 35 are effectively eliminated, although the "bump" in the mid-degrees is still evident. This pattern is

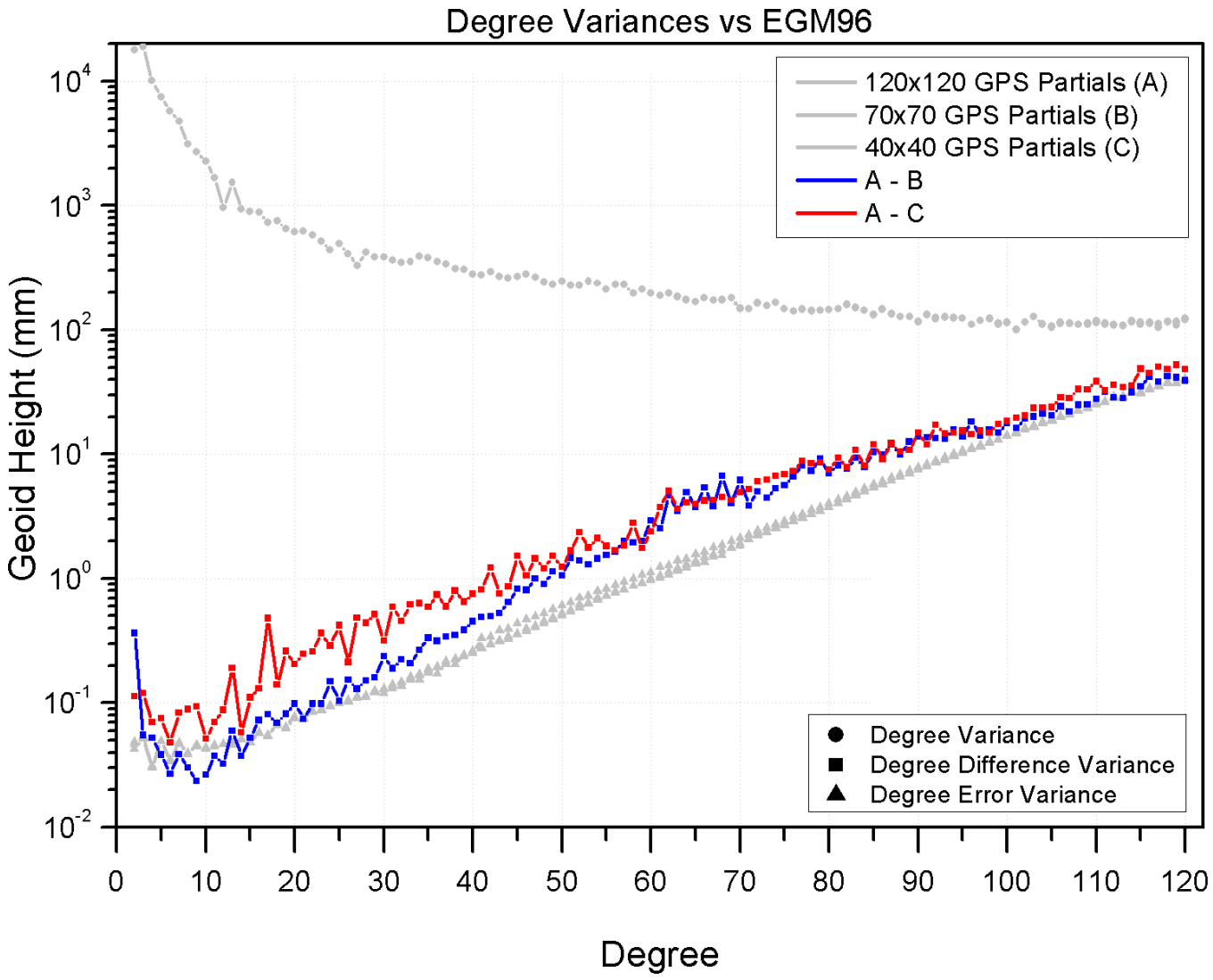


Figure 4.7: Degree difference variances for the April reduced GPS partials cases.

very similar to those seen in Figure 3.8 of the previous chapter, and indicates that errors of commission for the current GRACE data are large enough to impact the quality of the solutions if left untreated. However, this may only be a consequence of the current state of the GRACE data processing. As our understanding of the GRACE instruments and measurement data grows, the processing of the data may also improve to a point that the commission error is no longer significant.

From a processing efficiency standpoint, it is important to pursue the option of a reduced GPS parameterization. The use of even a 40x40 GPS parameterization, when compared to the full 120x120 scenario, would correlate into a 90% reduction in the size of the GPS data files created as well as a 75% reduction in compute time required to accumulate them. Fortunately, reducing the commission error through improved models is not the only method that will allow us to make use of a reduced GPS parameterization. As will be described in the next section, a technique in which the GPS weights are artificially reduced has proven to be an effective way to attenuate the effect of the commission errors in the presence of a reduced GPS parameterization.

4.7 GPS Downweighting

The results of the reduced network and reduced parameterization studies raised some questions with regard to the weights that the GPS measurements were being given. Since most of the gravity signal is recovered from the KBR data, the fact that even with the reduced network there are still three times as many GPS measurements as KBR measurements suggests that the GPS data may be dominating certain aspects of the gravity field estimation that it should not.

While the KBR and GPS data are optimally weighted with respect to each other, it was suggested by Watkins [70] that the resulting GPS data weights may still be too high in relation to the more accurate KBR data. To investigate this possibility, another series of 120x120 solutions were created from the same April data set in which the GPS weights were artificially downweighted by factors of 10 and 100 from their originally computed optimal weights. For this experiment, the GPS partials were set to 40x40, 70x70 and 120x120 and the twelve station reduced network was used. The degree variance plots of the 120x120 GPS partials case can be found in Figure 4.8. Although not shown, the results from the 40x40 and 70x70 case were quite similar to the 120x120 case when compared to EGM96. The plots show a slight improvement at the high degrees for both downweighted cases, along with an upturn of the formal errors at the low degrees. The addition of the formal error curve of GGM01C [60] (a 111 day GRACE gravity solution combined with surface information, and whose uncertainties were manually calibrated to more closely reflect the true errors in the solution) was done for comparison and lends support to the notion that this upturn is not necessarily undesirable, as the curves represent a more realistic error variance. On the whole, the solution differences are remarkably small considering the dramatic change in the GPS weights, indicating that the GRACE gravity solutions are dominated by KBR data, as we would expect. The total change in the solution as a result of the downweighting for the 120x120 case can be seen in Figure 4.9. The figure indicates that there is a small, but consistent difference between the downweighted cases and the original case. Unfortunately, the plot only shows that there was a change in the solutions, and doesn't necessarily demonstrate that the change was good or bad. The

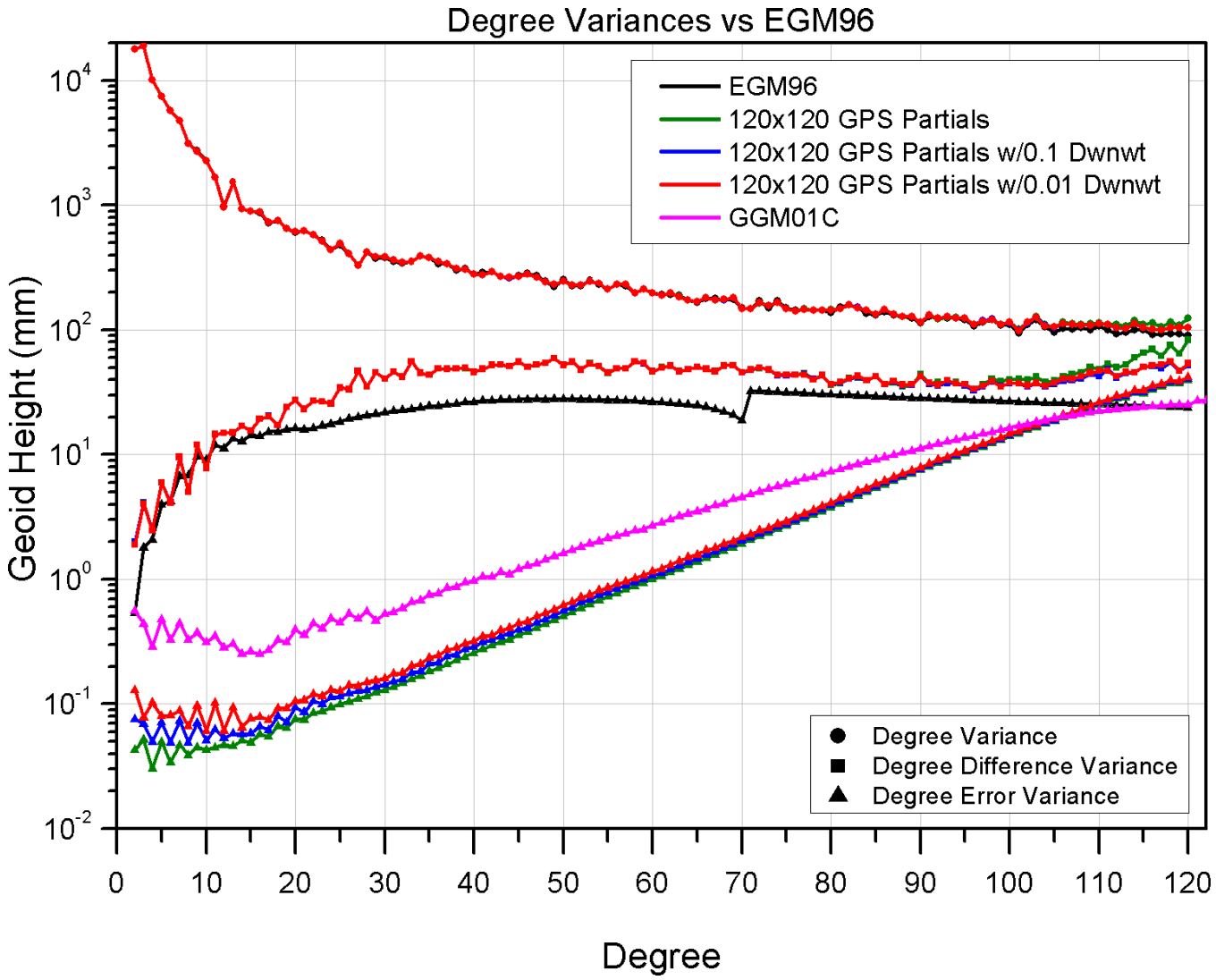


Figure 4.8: Solution of the April 120x120 GPS partials case downweighted by factors of 10 and 100 relative to their originally computed optimal weight. A slight improvement can be seen at the high degrees as well as a noticeable upturn at the low degrees. GGM01C is added for comparison to illustrate that this upturn is not necessarily bad, and may actually represent a more realistic error variance.

		Downweighting Factor								
		None			0.1			0.01		
Satellite	**1/rev**	**120**	**70**	**40**	**120**	**70**	**40**	**120**	**70**	**40**
GEOS-3		6.8	6.8	6.9	6.8	6.8	6.8	6.8	6.8	6.8
GFZ-1		34.6	34.7	34.5	34.6	34.6	34.6	34.5	34.5	34.6
GFZ-1	yes	7.8	7.8	7.8	7.8	7.8	7.8	7.8	7.8	7.9
Lageos 1		1.09	1.14	1.09	1.11	1.13	1.12	1.11	1.11	1.13
Lageos 2		1.03	1.1	1.07	1.05	1.07	1.06	1.04	1.04	1.07
Starlette		8.4	9.6	8.5	8.8	9.3	8.9	8.6	8.8	8.9
Starlette	yes	2.8	2.8	2.8	2.7	2.7	2.7	2.7	2.7	2.7
Stella		8.7	9.4	8.5	8.7	9.0	8.7	8.6	8.6	8.7
Stella	yes	3.0	3.0	2.8	2.9	2.8	2.8	2.8	2.8	2.8
Westpac		7.9	8.2	7.8	7.9	8.0	7.8	7.9	7.8	7.7
Westpac	yes	5.1	5.2	5.3	5.0	4.9	4.8	5.0	4.9	4.7

Table 4.4: Orbit test results for the April downweighting experiments using a select group of satellites. The 120, 70 and 40 column headings represent the maximum range of the GPS partials data. Table numbers represent RMS values in units of cm.

orbit and ocean circulation tests for the various downweighting solutions, listed in Tables 4.4 and 4.5, show a slight advantage to the downweighted cases, particularly for the 0.1 solutions.

4.7.1 Simulated Downweighting

To verify the apparent benefit of downweighting, the same procedure outlined above was applied to a set of simulated data. The simulation setup was identical to that described in Section 3.2. The force model resolution was set to 200x200, with the KBR partials fixed at 120x120, and the GPS partials ranging from 40x40 to 120x120. For each GPS partials range, a solution was run in which the GPS data was downweighted by factors of 10 and 100.

The results for the 120x120 GPS partials case can be seen in Figures

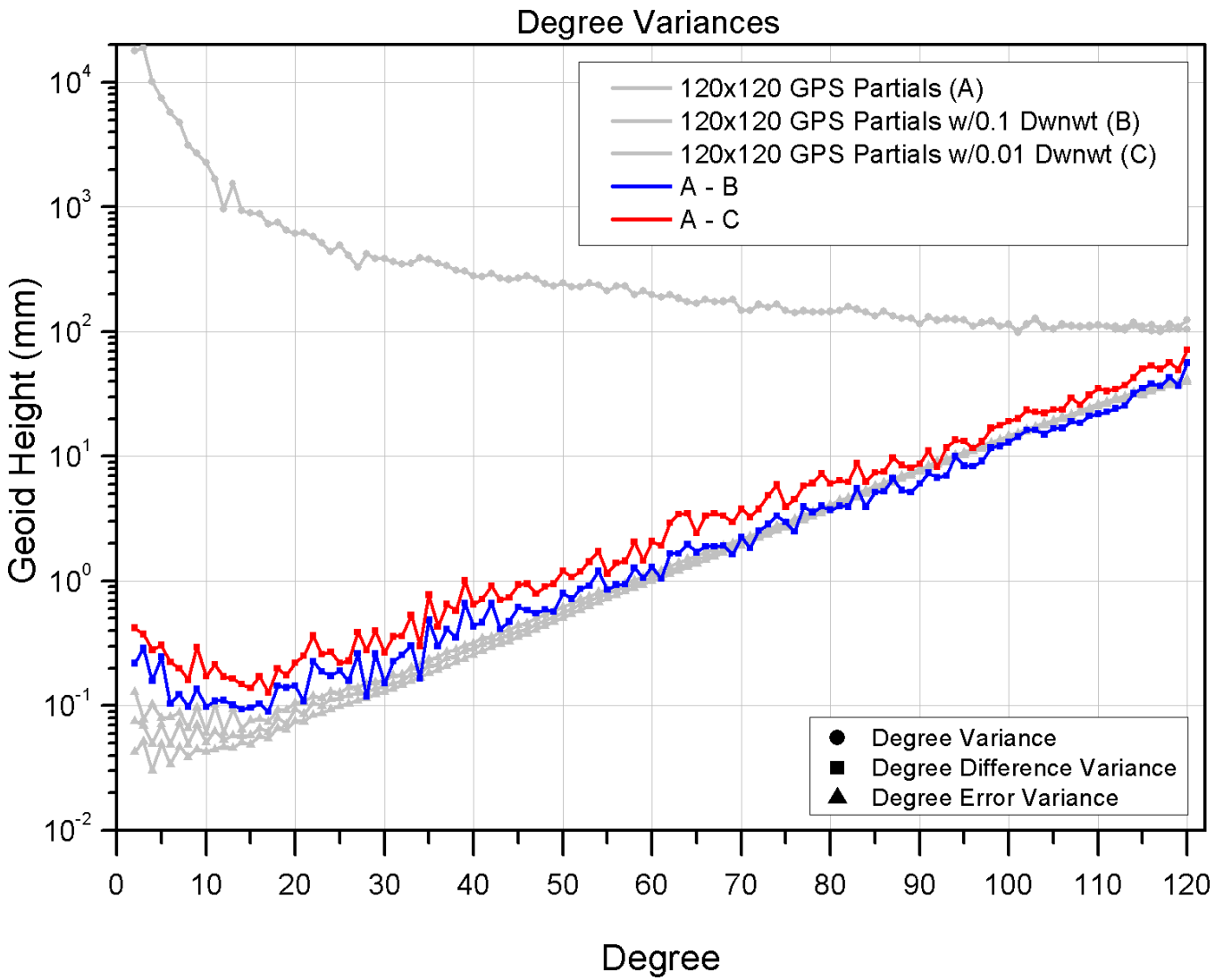


Figure 4.9: Degree difference variances for the various April 120x120 GPS partials cases with downweighting applied. The curves show a small, but consistent change is created as a result of downweighting.

Dwnwt Factor	GPS Range	Zonal RMS (cm/sec)	Zonal Correlation	Merid RMS (cm/sec)	Meridional Correlation
None	120	2.551	0.932	3.043	0.510
	70	2.586	0.930	3.135	0.487
	40	2.569	0.931	3.178	0.476
0.1	120	2.559	0.931	3.009	0.520
	70	2.578	0.930	3.036	0.513
	40	2.568	0.931	3.081	0.503
0.01	120	2.564	0.931	3.050	0.510
	70	2.571	0.930	3.059	0.508
	40	2.566	0.931	3.094	0.499

Table 4.5: Ocean circulation statistics for the April 120x120 GPS partials downweighting experiments.

4.10 and 4.11. In Section 3.6.1, the 120x120 GPS partials case performed best in terms of reducing the commission error, and produced the most accurate field of the simulations tested. Differencing the downweighted cases against the known truth (see Figure 4.11) shows a small, but consistent improvement for both the 0.1 and 0.01 downweighting. This is an important finding, because it shows that the full 120x120 GPS partials case can still be improved upon through the use of downweighting. It also shows that the "turn-up" at the low degrees is not an indication that the solution is getting worse at those degrees. In fact, the blow-up panel of Figure 4.11 shows the solution with the 0.1 downweighting to be more accurate than the non-downweighted case for nearly all degrees.

Having demonstrated that downweighting can improve the full 120x120 case, the next task was to evaluate the effect of downweighting a reduced partials case. As seen in Figure 4.12, the results were encouraging when compared to the non-downweighted full 120x120 GPS partials case. The 40x40

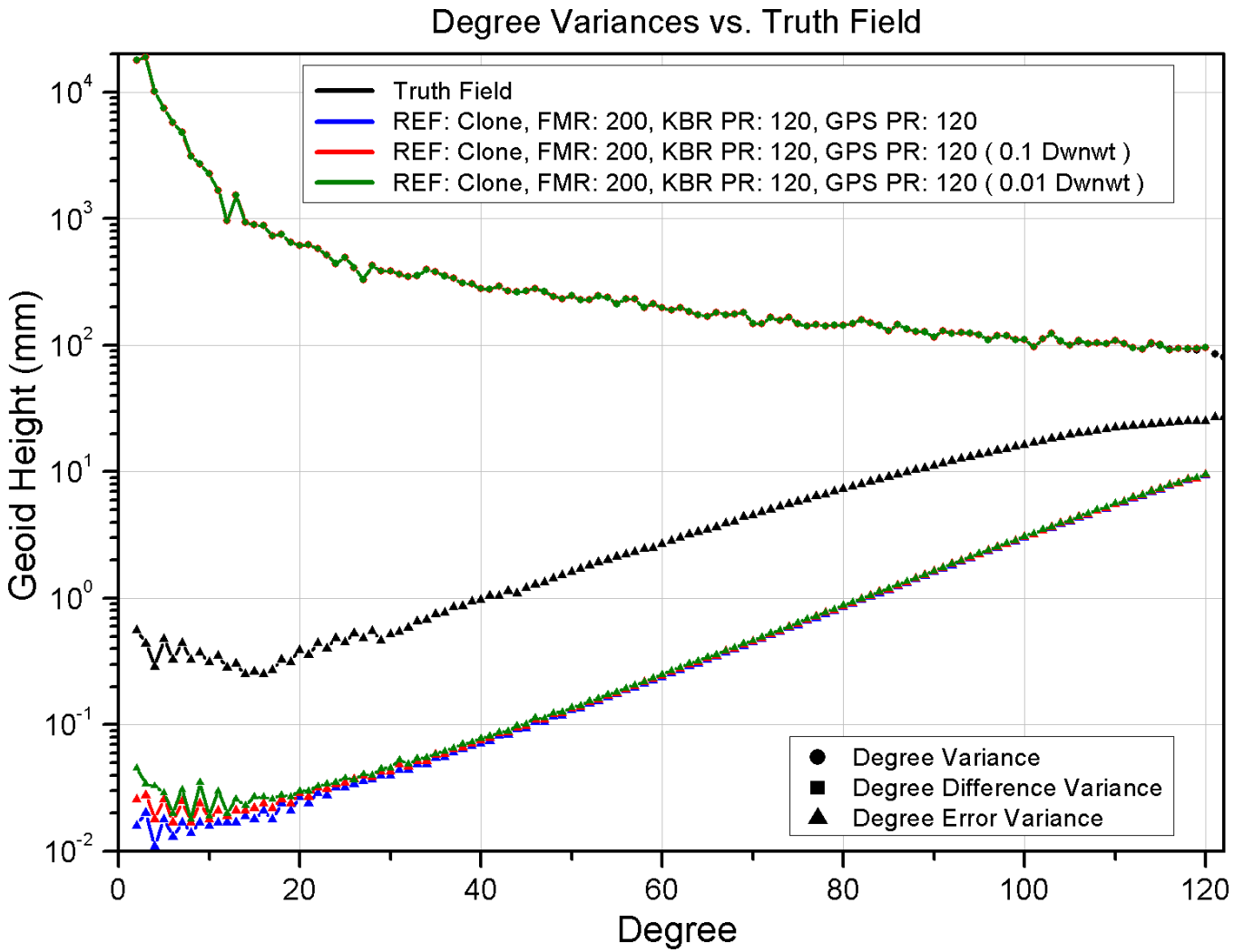


Figure 4.10: Degree variance plots for the simulated downweighted 120x120 GPS partials case.

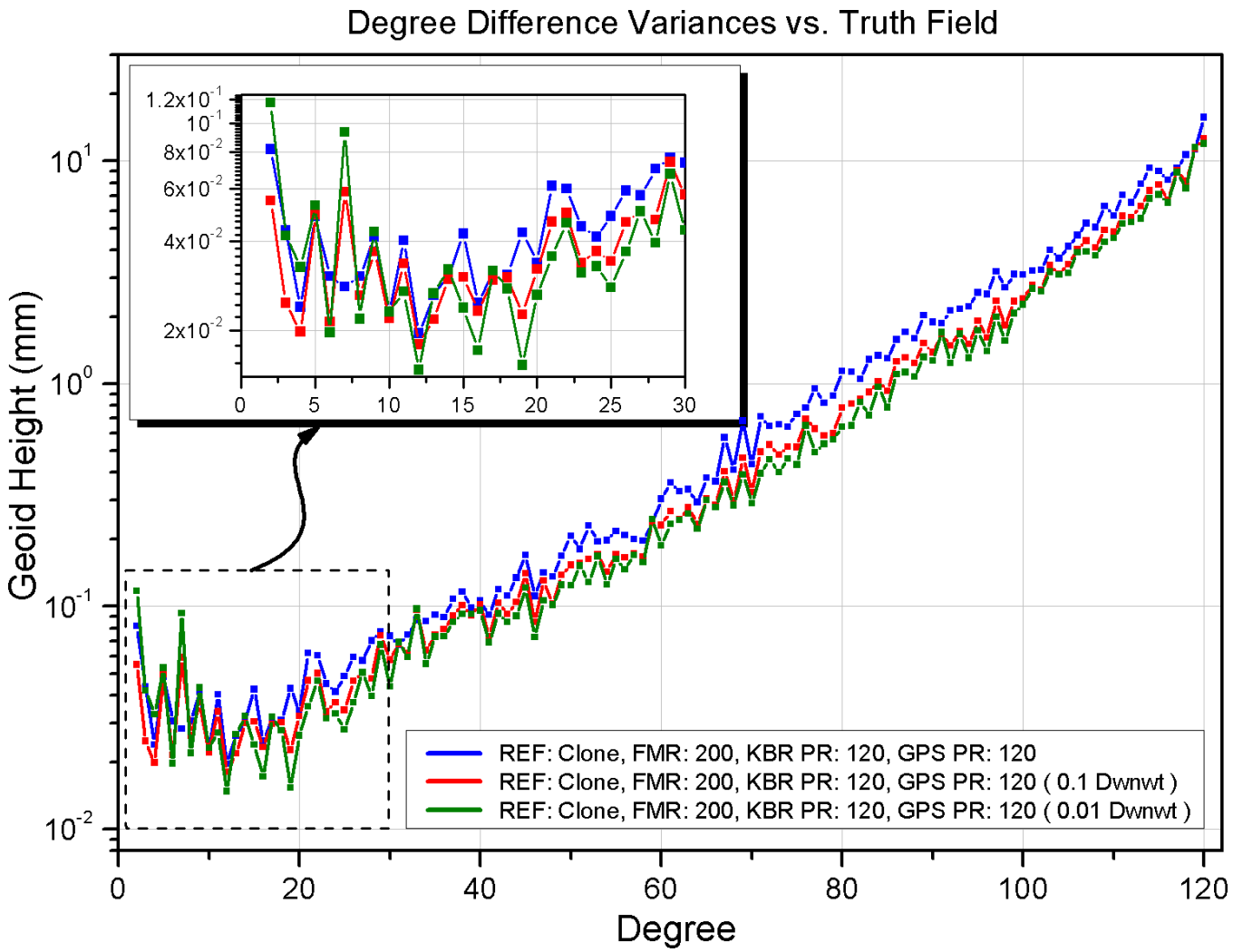


Figure 4.11: Degree difference variance plots for the simulated downweighted 120x120 GPS partials case. The 0.1 downweight case performs better than the non-downweighted case at nearly all degrees.

downweighted degree variances were closer to the truth field than the original 120x120 GPS partials case for nearly all degrees. In addition, when the 40x40, 0.01 downweighted case was compared to the 120x120, 0.01 downweighted case, the two solutions were nearly identical (see Figure 4.13). Both of these figures represent another significant finding, because they support the notion that a reduced GPS partials solution that employs downweighting can be used to achieve an equivalent or better solution than a full 120x120 GPS partials downweighted or non-downweighted case.

4.8 Conclusions

The experiments of this chapter addressed some concerns regarding the potentially overwhelming number of GPS observations available when generating GRACE gravity fields. Several strategies were developed which greatly reduce the number of GPS observations involved in the estimation process without affecting the overall quality of the gravity field models.

Using a reduced network of twelve carefully chosen GPS ground stations did not significantly degrade the quality of the gravity field estimates. In fact, the solutions showed a slight improvement at the high degrees and at C20. Decreasing the number of ground stations greatly improves processing efficiency by reducing the total number of double-differenced GPS observations by roughly 67 %.

Decreasing the parameterization of the GPS data does improve the determination of J2, but does so at the cost of degrading the mid-degrees. A 40x40 and 70x70 reduced GPS parameterization test case showed similar behavior as some of the simulations performed in Chapter 3, suggesting that the degra-

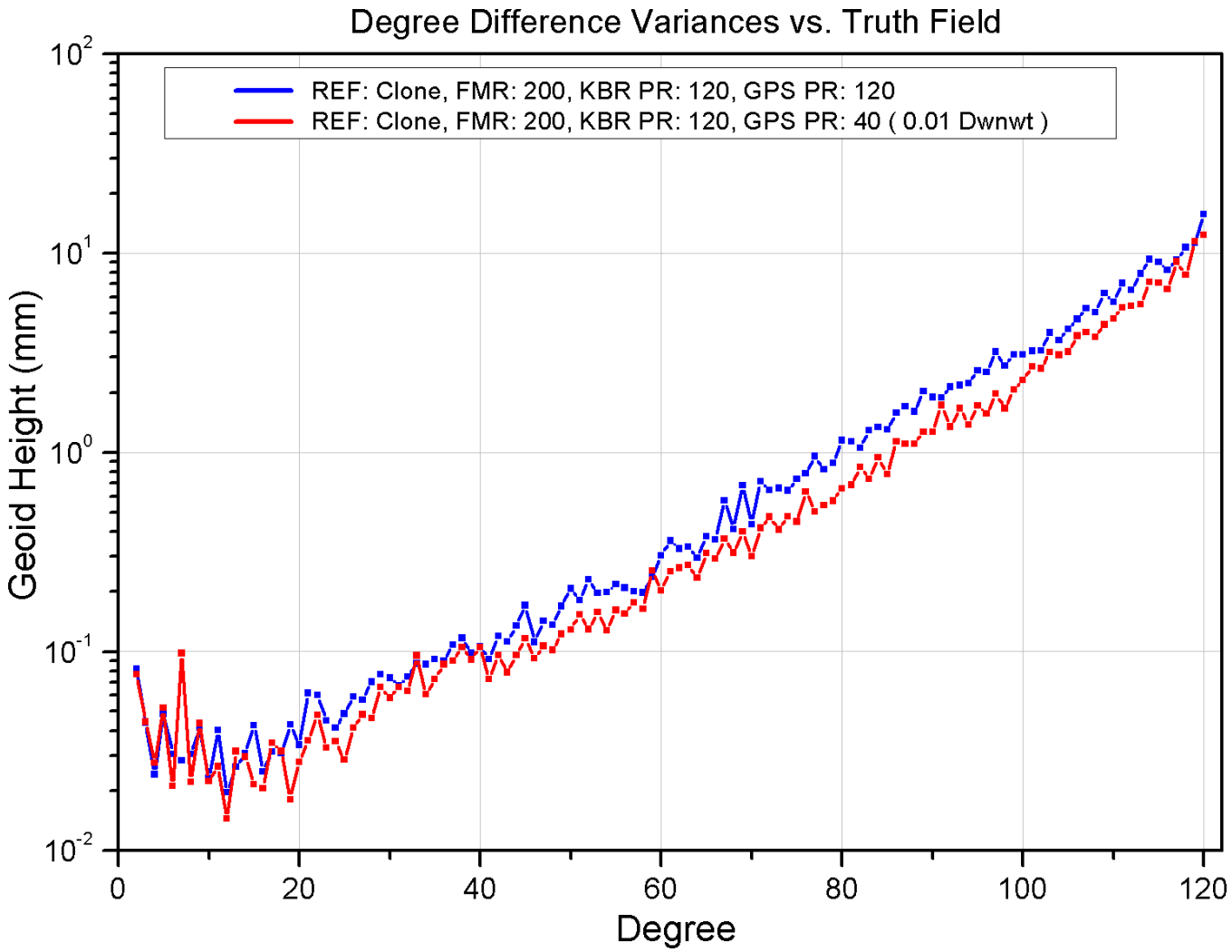


Figure 4.12: Degree difference variance plots for the simulated 120x120 GPS partials case and the downweighted 40x40 GPS partials case. This figure illustrates how a reduced partials case can outperform an extended partials case by employing downweighting.

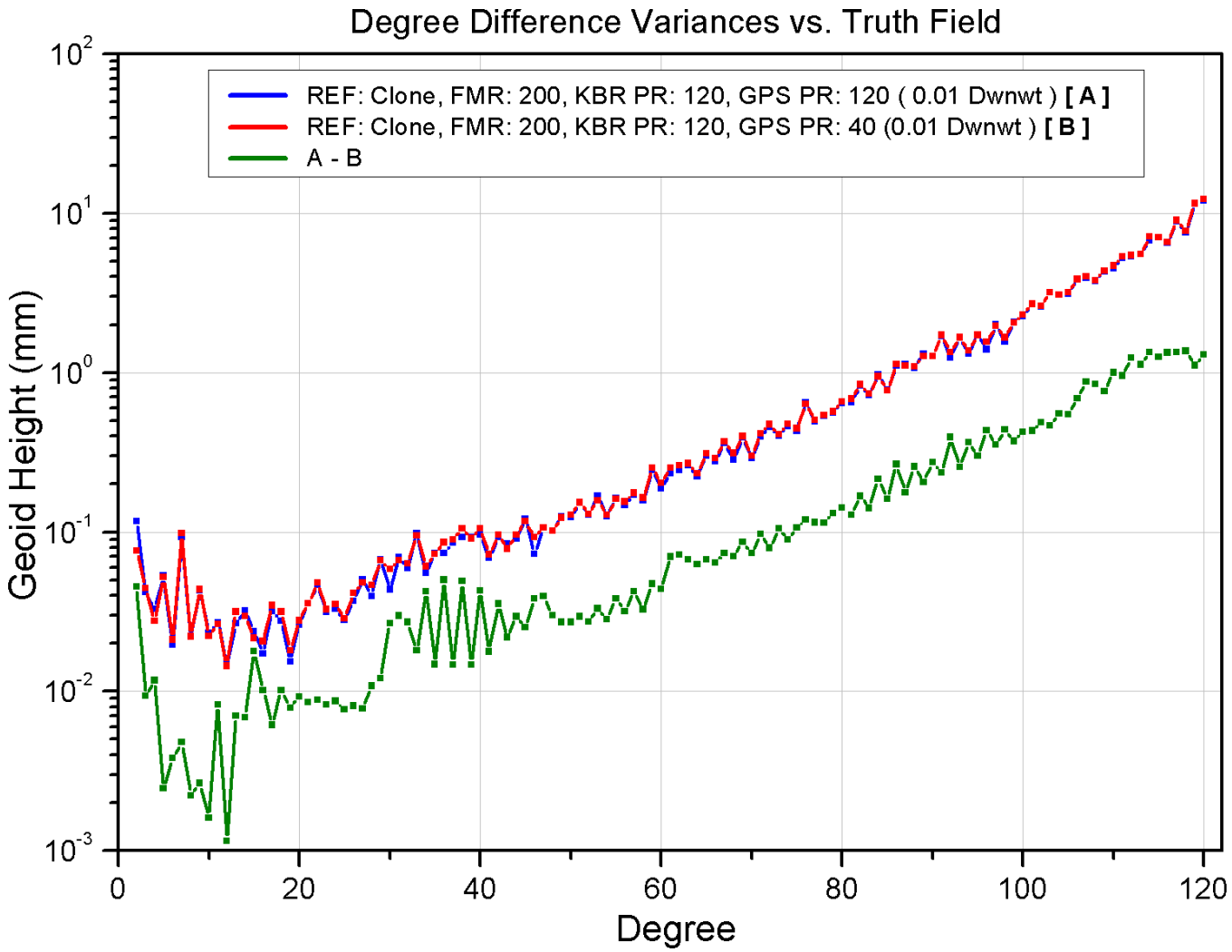


Figure 4.13: Degree difference variance plots for the simulated downweighted 40x40 and 120x120 GPS partials cases. The small difference between these solutions shows there is no advantage to using an extended GPS partials set when downweighting is applied.

dation at the mid-degrees was due to the influence of errors of commission. Fortunately, a procedure in which the GPS data is manually downweighted from its computed optimal weight greatly reduces the impact of this commission error. Using this technique, a downweighted 40x40 reduced GPS partials solution performs as well or better than a non-downweighted, full 120x120 GPS partial case.

Combining the techniques outlined in this study, i.e., using the reduced network, along with a 40x40 reduced parameterization downweighted by a factor of 10 or 100, created a measurable improvement in the quality of the resulting gravity field model. In addition, the findings of this study have cut the processing time and disk storage requirements for an average solution by roughly 75 %.

Chapter 5

Conclusions

5.1 Summary and Conclusions

The goal of this study was to explore the impact that various error sources and processing strategies have on the gravity field models created from the GRACE RL01 mission data. Due to the high precision and unique characteristics of the GRACE instruments, many of the traditional processing choices, particularly those involving the least squares estimation phase, were re-evaluated to ensure the recovery of the most accurate gravity field models possible. The first step towards achieving this goal involved the creation of a parallel least squares solver designed to accommodate the large volume of equation sets that are created from the GRACE mission data. This new software tool was an essential part of the work done for this study, enabling the routine solution of high degree and order gravity field models from both real and simulated data.

When this work first began, the need to routinely solve large linear systems involving tens of thousands of parameters using terabytes worth of data was one of the many challenges that the GRACE mission presented (See Appendix B.1). The legacy software at the time was more than capable in terms of functionality, but it was not designed to handle the extremely large data sets that the GRACE mission would produce. To accommodate these needs, the first effort of this work was devoted towards developing a least squares

solver that was designed to run in a parallel environment. The operations involved with the least squares process are well suited for massively parallel machines, in which tens or hundreds of processors can work simultaneously on a problem. While the software's core function would be to perform the least squares reduction, it also had to have many other complex features, such as the ability to solve parameters over varying intervals, compute covariances, employ shifting and optimal weighting, and much more. Over time, the software evolved into what is now known as the Advanced Equation Solver for Parallel Systems (AESoP). The details of the primary least squares algorithm used by AESoP are presented in detail in Chapter 2, along with a few other supporting algorithms. AESoP now serves as one of the primary processing tools for the GRACE mission, and has logged over 200,000 CPU hours to date.

A byproduct of the research behind AESoP involved the creation of a new out-of-core algorithm used to deal with problems that are too large to fit even in the combined memory of today's massively parallel machines. The out-of-core algorithm, in which the matrix being factored is stored on disk and processed incrementally, is a new and efficient alternative to traditional out-of-core solvers and enables the scalable solution of problems of an almost arbitrary size. The new out-of-core algorithm has already been used to solve for a preliminary 360x360 (30,000 parameters) gravity field [10], complete with full covariance. To the author's knowledge, this is the largest rigorous (i.e., without the use of approximation) field ever computed, and illustrates the power and potential of the out-of-core algorithm.

Having developed the tools necessary to create high degree and order gravity fields from GRACE data, the next step was to begin the investigation

into certain error sources present in the batch estimation process, such as the errors of omission and commission. These errors are nearly impossible to isolate in real-data analysis, so a series of simulations were constructed to observe the influence of these errors on the gravity solution in a controlled environment. The primary conclusion reached from these experiments was that, for the current RL01 GRACE processing scenario, the truncation of the GPS partials in the presence of commission errors created an artifact in the solutions that was large enough to be of concern. While this artifact could be sufficiently mitigated by using a full set of GPS partials, it carried the undesired byproduct of creating rather large GPS data files.

The results of these simulations were just one of the motivating factors for the investigation into the next topic of this study, which involved the combination of the GPS data with the inter-satellite range data. When a full GPS ground station network is used, and the GPS measurement partials are extended to the full range of the field being estimated, roughly 90% of the processing time required to generate a solution is spent on the GPS data. Since the sensitivity of the GPS data to gravitational perturbations is limited in comparison to the much more accurate K-band ranging measurements, it seemed unnecessary to spend the bulk of the processing time on the GPS data. A number of experiments, using both real and simulated data, were conducted that examined various methods of reducing the influence and number of GPS observations involved in the estimation process without compromising the quality of the resulting fields. By reducing the GPS ground network to twelve ground stations, it was concluded that the number of GPS observations involved in the solution could be dramatically reduced without significantly impacting the

field quality. In addition, a strategy involving the intentional downweighting of a truncated GPS data set was discovered to effectively mitigate the artifacts due to commission error observed earlier in the simulations of Chapter 3. Using a GPS data set truncated to 40x40, with downweighting applied, was shown to create a field that is nearly identical to the equivalent case in which the full range of GPS partials were used. When the smaller GPS ground network is combined with the downweighted reduced GPS partials data, the processing time and disk storage requirements for an average GRACE RL01 solution can be cut by roughly 75 % and with no measurable degradation in the quality of the resulting gravity field models.

In conjunction with recent hardware advances, the software tools created as part of this work have increased the speed and processing power available to researchers by an order of magnitude. The tools have enabled an in-depth analysis of several high frequency error sources that would not have otherwise been possible. The results of some of this analysis have already been adopted into the GRACE standard processing scheme.

5.2 Further Studies

There were a number of questions and potential research avenues created through the course of this study. The tile-based QR algorithm of the out-of-core solver alone opens up a wide range of possibilities. The tile-based approach has potential for a number of other similar dense linear operations. Ideally, a full suite of out-of-core utilities could be developed to allows scientists to solve large research problems on limited computing resources.

The relationship between the errors of commission and the GPS partials

could be explored in more depth. While it was observed that the GPS data is particularly sensitive to the errors of commission, it was not clearly understood why this was the case, or why it did not affect the KBR partials to the same degree. The reduced partials studies could also be extended to see if selected resonant partials could be used to achieve the same effect as estimating the full range of GPS partials.

The downweighting factors used in this work were limited to factors of 10 and 100. Based on the various field evaluations, it appeared that the optimal downweight factor might be somewhere in between this range. If downweighting is to be adopted as a permanent processing strategy, then more effort should be put into finding the optimal downweight factor.

Appendix

Appendix A

Estimation Theory

The following information is provided as a brief outline of the theory and techniques used in estimating the Earth's gravity field model. The sections below are a condensed description of the estimation problem behind the GRACE mission and are by no means a complete discussion of the topic. The reader is encouraged to consult the wide body of literature available on the subject [63, 58, 48] for further details.

A.1 The Geopotential Model

It can be shown [35] that the Earth's geopotential can be closely approximated through the use of spherical harmonics by the expression

$$U(r,\phi,\lambda,t) = \frac{\mu}{r}\sum_{l=2}^{\infty}\sum_{m=0}^{l}\left(\frac{a_e}{r}\right)^l \overline{P}_{lm}(sin\phi)[\overline{C}_{lm}(t)\cos m\lambda + \overline{S}_{lm}(t)\sin m\lambda] \tag{A.1}$$

where λ, ϕ and r represent the spherical coordinates at which the potential is evaluated. The quantities l and m represent the degree and order of the spherical harmonic expansion, and $\overline{P}_{lm}$ are the normalized associated Legendre polynomial functions. The constants μ and a_e represent the geocentric gravitational constant and equatorial radius of the Earth. The normalized spherical harmonic coefficients $\overline{C}_{lm}$ and $\overline{S}_{lm}$ represent the mean and time variable components of the potential. The above expression also assumes an Earth-fixed

reference frame. As illustrated in Fig. A.1, the gravity coefficients are typically categorized into three main classes: zonals, tesserals and sectorials. All

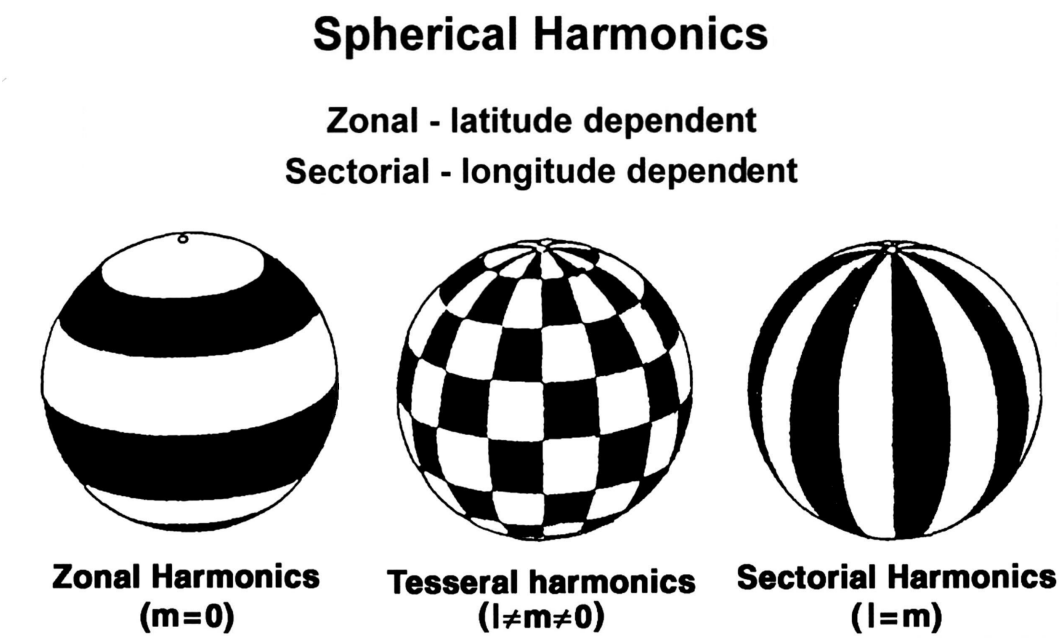


Figure A.1: An illustration of spherical harmonics.

components of equation A.1 can be determined accurately with the exception of the terms relating to the Earth's density (i.e., $\overline{C}_{lm}$ and $\overline{S}_{lm}$). One method of determining the value of these coefficients is to measure the path of an orbiting satellite and compare this against its computed, or predicted, path as determined by a reference gravity field model.

Direct measurements of a satellite's position are usually in the form of range and range-rate observations, as taken from satellite laser ranging (SLR) systems or other ground based tracking systems. In addition to SLR data, GRACE relies on GPS data (both pseudorange and phase measurements), as

well as the KBR tracking data, to determine each satellite's position and velocity in space. Since the position and velocity are not measured directly, a second model needs to be created, called the measurement model, that relates the observation data (i.e., range, range-rate, GPS, KBR, etc.) to the dynamical model of equation A.1. If the models are correct, then the computed orbit of the satellite should come very close to its actual observed path.

A.2 Least Squares Estimation

The involvement of least squares estimation with the creation of gravity field models is straightforward. We begin by representing the dynamic and observation models as functions of the state, $X(t)$. [1]

$$\begin{array}{lcll} \dot{X}(t) & = & F(X(t),t) & \text{Dynamic Model} \\ Y & = & G(X(t),t)+\epsilon & \text{Observation Model} \end{array}$$

where ϵ is an error term caused by factors such as measurement noise, model errors, and numerical errors.

The next step is to create a reference, or nominal, trajectory that is close enough to the true trajectory to allow us to linearize the equations of the dynamical and observation models. We are not directly estimating the values of the model coefficients, $X(t)$, but rather the *deviation* between the true and nominal values. These deviations, or residuals, of the state and observations are defined as

$$\begin{aligned} x &= X - X^* \\ y &= Y - Y^* \end{aligned}$$

[1] $X(t)$ is called the *state vector* and contains model parameters such as position, velocity, geopotential coefficients, etc.

where the * denotes the nominal values. Expressing y in terms of the observation model, we have

$$y = g(X(t),t) - g(X^*(t),t) + \epsilon$$

Since we have assumed that the nominal trajectory is within close proximity to the true solution, we can expand y about X^* via Taylor expansion,

$$\begin{aligned} y &= g(X^*(t)+x(t),t) - g(X^*(t),t) + \epsilon \\ &= g(X^*(t),t) + \left(\left[\frac{\partial g(X(t),t)}{\partial X(t)}\right]_{X=X^*}\right)x(t) + \text{h.o.t.} - g(X^*(t),t) + \epsilon \\ &= \left(\left[\frac{\partial g(X(t),t)}{\partial X(t)}\right]_{X=X^*}\right)x(t) + \epsilon \\ &= \tilde{H}(t)x(t) + \epsilon \end{aligned} \tag{A.2}$$

where $\tilde{H}(t) = \left[\frac{\partial g(X(t),t)}{\partial X(t)}\right]_{X=X^*}$. Taking the time derivative of the state deviation, x, and performing a similar expansion about X^*, we obtain the linear system

$$\dot{x} = A(t)x$$

with $A(t) = \left[\frac{\partial F(X(t),t)}{\partial X(t)}\right]_{X=X^*}$. The solution to this system is

$$x(t) = \Phi(t,t_0)x_0 \tag{A.3}$$

where t_0 is some specified epoch and $\Phi(t)$, the state transition matrix, satisfies the following

$$\dot{\Phi}(t,t_0) = A(t)\Phi(t,t_0) \quad \text{and} \quad \Phi(t_0,t_0) = I.$$

By substituting A.3 into A.2 we arrive at the expression

$$y(t) = H(t)x_0 + \epsilon$$

with $H(t) = \tilde{H}(t)\Phi(t, t_0)$. Assuming there are m observations, the m data equations may be written in matrix form as

$$y = Hx_0 + \epsilon \tag{A.4}$$

where the $m \times n$ matrix H is known as the *partials matrix*. To solve for the state residuals, x_0, requires the least squares solution of the linear system of Eqn. A.4. This solution is called the *estimate* and is represented by $\hat{x}_0$, or simply $\hat{x}$. Once the estimate, $\hat{x}$ is found (through QR or Normal Equations), it is added to X^* to obtain the final estimated model parameters,

$$\hat{X} = X^* + \hat{x}.$$

A.3 Optimal Weighting

The process of estimating a gravity field model involves the incorporation of many data points collected from a number of different instruments. The accuracy and precision of these instruments are not always equal. An example of this, in terms of the GRACE mission, can be seen in the difference between the GPS positioning data and the K-band inter-satellite ranging measurements that are collected from both of the GRACE satellites. The K-band antennas were designed to be extremely precise, recording range measurement accurate to less than 10 micrometers, while the GPS data is much more coarse at the centimeter scale precision. The two data types each have their place in the modelling process, but in terms of the sensitivity to the Earth's gravitational potential, the KBR data is the primary observable for the GRACE mission. Therefore, it is important that the less sensitive GPS data not be given the same value when estimating the gravity field.

Even within a given data type, various instrument errors or temporary malfunctions can cause one group of measurements to be less accurate than another. Even though one group is of lower quality than another, each data set still has some contribution to the estimates of the given parameter set. To combine these data sets of mixed quality, we would want to give more weight the observations of higher "quality" and lower weight to those of lesser "quality".

One approach that has been developed by Tapley et al. [61, 64], and later expanded by Yuan [72], is to examine the post-fit residuals of each data set and assign a weight (i.e., quality assessment) based on how well the computed estimates fit the data. It begins with the definition of a performance index

$$J(x) \;=\; (y - Hx)^T W (y - Hx)$$

where W is a weight matrix. Because AESoP and the least squares algorithm it employs make use of the QR factorization instead of the traditional normal equations approach, the performance index, J, has the equivalent form

$$\begin{aligned}
J(x) &= (y - Hx)^T W (y - Hx) \\
&= (y - Hx)^T W^{T/2} Q Q^T W^{1/2} (y - Hx) \\
&= (Q^T W^{1/2} y - Q^T W^{1/2} Hx)^T (Q^T W^{1/2} y - Q^T W^{1/2} Hx) \\
&= (\begin{bmatrix} b \\ e \end{bmatrix} - \begin{bmatrix} R \\ 0 \end{bmatrix} x)^T (\begin{bmatrix} b \\ e \end{bmatrix} - \begin{bmatrix} R \\ 0 \end{bmatrix} x) \\
&= (b - Rx)^T (b - Rx) + e^T e
\end{aligned}$$

Once a set of estimates, $\hat{x}$, has been computed then the value of J becomes

$$J(x) \;=\; J(\hat{x}) \;=\; (b - R\hat{x})^T (b - R\hat{x}) + e^T e$$

And a new weight can be calculated for the given data set

$$f \;=\; m / J(\hat{x})$$

where m is the number of observations in the data set.

A.4 Shifting

When combining two or more sets of linearized equations, it is important to make sure that the corrections that get computed from the least squares reduction process are applied to the same nominal field. If the nominal field used to evaluate the measurement partials differs from one data set to the next, a technique called *shifting* must be applied to the equations so that the data refers to the same nominal. The technique is straightforward, with the assumption that the two nominal fields are relatively "close" to one another.

Using the notation described in Section A.2, we will assume that two data sets exist that have been evaluated with different nominal fields

Data Set 1:

$$\begin{aligned} y_1 &= H_1 x_1 + \epsilon_1, \text{ using Nominal } X_1^* \\ x_1 &= X - X_1^* \end{aligned}$$

Data Set 2:

$$\begin{aligned} y_2 &= H_2 x_2 + \epsilon_2, \text{ using Nominal } X_2^* \\ x_2 &= X - X_2^* \end{aligned}$$

and that we wish both data sets to reference, or shift to, the nominal field of data set 1 (i.e., X_1^*). To do this, we take the equations for data set 2 and apply

the following

$$\begin{aligned} y_2 &= H_2 x_2 + \epsilon_2 \\ &= H_2(X - X_2^*) + \epsilon_2 \\ &= H_2(X - X_1^* + X_1^* - X_2^*) + \epsilon_2 \\ &= H_2(x_1 + X_1^* - X_2^*) + \epsilon_2 \\ &= H_2 x_1 + \epsilon_2 + H_2(X_1^* - X_2^*) \end{aligned}$$

Moving some quantities to the other side of the equation and defining the term

$$\Delta X = X_2^* - X_1^*,$$

we have

$$\begin{aligned} H_2 x_1 + \epsilon_2 &= y_2 - H_2(X_1^* - X_2^*) \\ &= y_2 + H_2(X_2^* - X_1^*) \\ &= y_2 + H_2 \Delta X \end{aligned}$$

By making an appropriate adjustment to the *observation* residual, y_2, we see that data set 2 can be treated with respect to nominal field of data set 1. If the two data sets share a common set of parameters, this algorithm allows both data sets to be combined and to contribute to the estimation of those parameters.

An example of when the shifting procedure is applied can be seen when a set of parameters are estimated across multiple arcs, i.e., "super-arced" common parameters. If a set of data files was created using a one day arc length, then the parameters in each file will have an *a priori* value corresponding to

the start of each day. If an experiment is conducted in which a set of parameters is estimated over more than one day, then the observations in each file of the multi-day set would need to be shifted so that the super-arced parameters all reference the same nominal field (typically those of the first day in the multi-day set). Common examples of this in the GRACE data processing scheme would be parameters such as the accelerometer scale factors, which are typically estimated over multiple days.

Appendix B

GRACE Processing Scheme

The following information is intended to give the reader a brief overview of the data and procedures of the GRACE RL01 processing scenario. The complete data processing standards and a user handbook for the GRACE gravity solutions are available at the following URL:

`www.csr.utexas.edu/grace/publications/handbook`

B.1 General Processing Flow

The collection and processing of the GRACE mission data is a complex process involving a number of different agencies located across the globe. The tasks are divided into primary systems: Mission Operations Systems (MOS) and the Science Data Systems (SDS). A majority of the MOS duties are conducted by the German Space Operations Center (GSOC), and involve the maintenance of the satellites and the collection of the raw science data. One component of the SDS manages the initial processing of the science data (star camera, accelerometer, K-band ranging information, etc.), including basic clean-up of the data as well as packaging the data into predefined file formats. Other components involve the interpretation and combination of these data files into science products (i.e., gravity field models), which is the primary focus of this study.

The processing scheme described here is specific to the GRACE RL01 data processing scenario. It begins with the assumption that a series of GPS double-differenced observations (sampled at 30 second intervals) and K-band range-rate observations (sampled at 5 second intervals) have been created. Based on these observations, a set of measurement partials are created for each data type based on the process outlined in A.2, using a batch integration period of one day. This process makes use of the Multi-Satellite Orbit Determination Program (MSODP) [46] and involves the numerical integration of hundreds of thousands of partial differential equations. The equation sets contained in the daily GPS and KBR partials files are then accumulated using AESoP using an orthogonal least squares reduction procedure. This last step is what actually creates the estimates of the gravity coefficients that comprise a gravity field model. Typically, a month's worth of GPS and KBR data are combined in a single run to create monthly gravity models. These monthly accumulation can in turn be combined to create semi-annual or annual solutions, as illustrated in Figure B.1 illustrates. Depending on the parameterizations used, as well as the sampling rate of the data, the computational resources needed to create a solutions can vary substantially.

B.1.1 Computational Requirements: An Example

The GRACE mission data is expected to support a maximum spatial resolution of roughly 250 k, translating into a 160x160 gravity field model. A field of this size involves roughly 26,000 (n) parameters. To create a single day's worth of measurement partials for a field of this size is an $O(6n)$ operation, requiring the numerical integration of $\sim 156,000$ differential equations. The combined

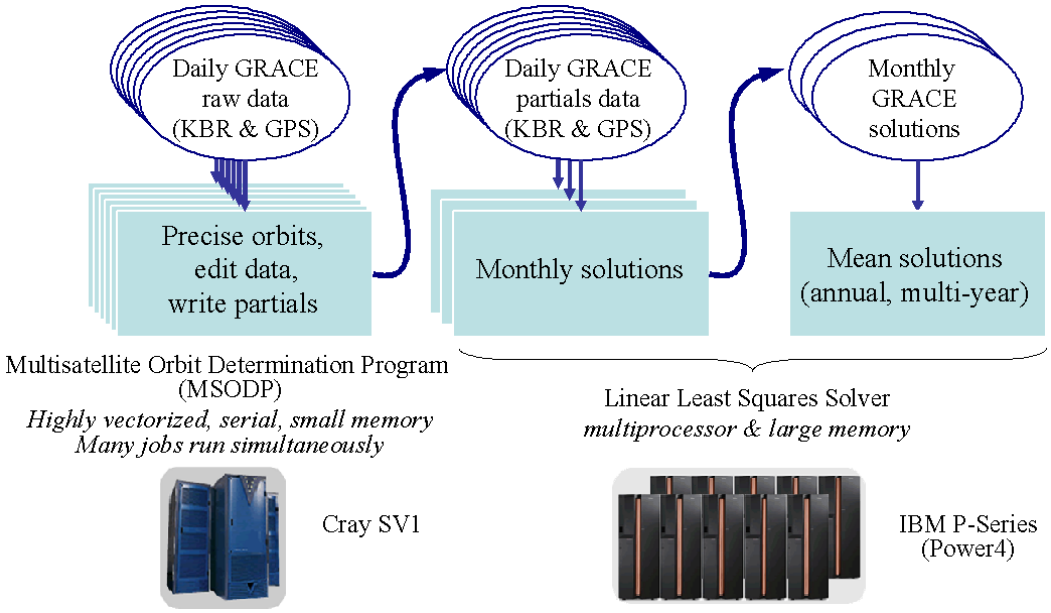


Figure B.1: An outline of the GRACE data processing flow

size for a typical daily set of KBR (16,000 observations) and GPS (50,000 observations) partials files is $\sim$ 5 gigabytes. To accumulate these partials involves $O(mn^2)$ operations, where m is the total number of daily observations, in this case 66,000. This translates into over 21 trillion floating point operations (FLOPS). These statistics are only those required to process a single day's worth of data, and can easily increase to much larger totals as the solution time spans reach into the monthly and annual time frames. These numbers do not include additional storage requirements for items such as covariance files, accumulated R files (from the QR factorization), and others. As this example illustrates, the need for high performance computing and large volume data storage was of primary concern in the GRACE mission planning.

Appendix C

Simulation Details

The purpose of this appendix is to supply the reader with more detailed information regarding the simulations used in this study. As mentioned earlier, much of the theory, development and application of the simulations conducted in this study followed directly from earlier work done by Kim [37], parts of which are repeated below for completeness. The reader is encouraged to read Kim's work for a much more in-depth explanation of the measurement noise modelling and parameterization choices employed in this study.

C.1 Simulation Procedure

The following is the step-by-step procedure followed to create the simulations used in this study.

1. Choose a truth field. This gravity field serves as the absolute truth in the simulation universe, and is the field that all results are compared against. Because of the controlled environment of a simulation, the choice of truth field is arbitrary, but should remain consistent when creating multiple data sets that will be compared to each other. For the purposes of this study, GGM01C [60] was chosen because it represented the best known gravity model at the time the simulations were made. GGM01C is complete out to 200x200, so EGM96 [40] was patched on to fill the truth

field out to 360x360. GGM01C and EGM96 relied on the same surface gravity information to compute their high degree and order components, so the inconsistency introduced by this patching is minimal.

2. Create a clone model. The clone model refers to a model that is intentionally different than the truth field, and is done to introduce modelling error into the simulations. The modeling errors are introduced to the clone by first performing a Cholesky ($A = LL^T$) factorization of the covariance matrix of the truth field. The matrix, L, is then multiplied by a normally distributed vector of random numbers with zero mean and a standard deviation of one. The result of this matrix-vector operation is then added to the truth field coefficients to create the clone. For parameters above 200x200, the covariance is not available, so the errors were created by applying a similar approach to the uncertainties of those coefficients. By creating the clone in this manner, the coefficient differences between the truth and clone fields should be within the level of the uncertainty of the truth field (i.e., 1-σ). Note that only the coefficients change in the cloning procedure, with the variances remaining the same. Figure C.1 shows the difference between the truth and clone fields used in all of the simulations presented in this work, in terms of square-root degree difference variance.

3. Generate the observations. A set of simulated observations are created based on the truth model (not the clone) using MSODP [46]. Simulated measurement noise is also added to the observations at this point. Non-gravitational perturbations such as solar radiation pressure, atmospheric drag and Earth radiation pressure are also added.

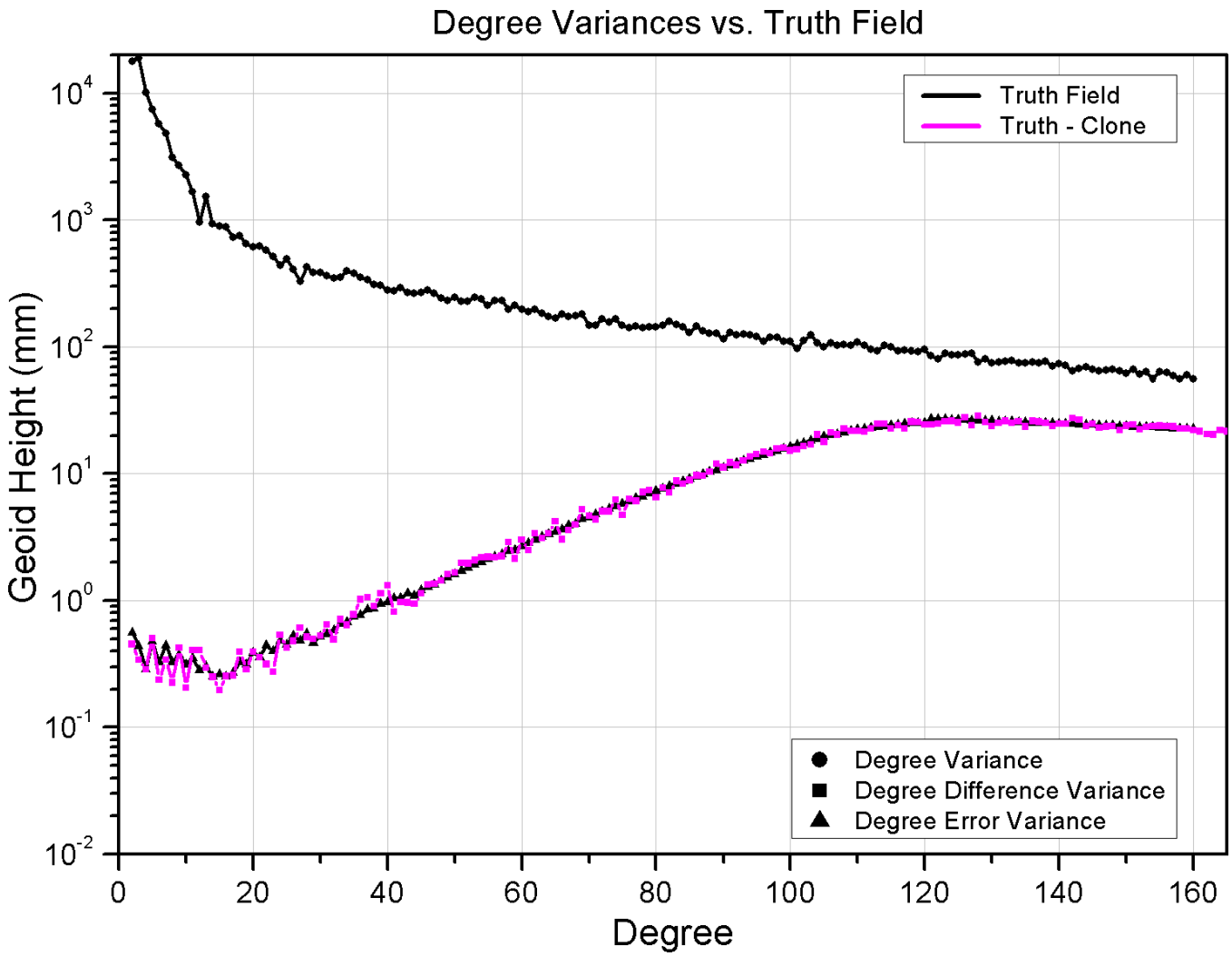


Figure C.1: Comparison of truth and clone reference fields.

4. Generate the measurement partials. Using MSODP once more, the partials for both the GPS and KBR data sets are then created using either the truth or clone model, depending on the experiment. If modeling errors are desired in the simulation, then the clone field should be used.

5. Converge the GPS orbits. To avoid the possible divergence of the linear system, the estimates of the GPS satellite orbit initial conditions are first computed. During this phase, the gravity coefficients are frozen so that only the initial conditions are adjusted. This helps to ensure that the initial conditions of the GPS orbits converge, or stay within the linear range required by the estimation process [52].

6. Least squares solution. Perform the least squares estimation of the linearized equation sets (i.e., the partials and observations) using AESoP. Typical solution options include *a priori* conditioning, optimal weighting and shifting (see Appendices A.3 and A.4).

C.2 Measurement Errors

In addition to the modeling errors that can be introduced through the use of the clone field, a series of standard measurement errors were introduced to various components of the simulations to better replicate the GRACE RL01 processing environment. They include errors introduced to the GPS, KBR and accelerometer measurements.

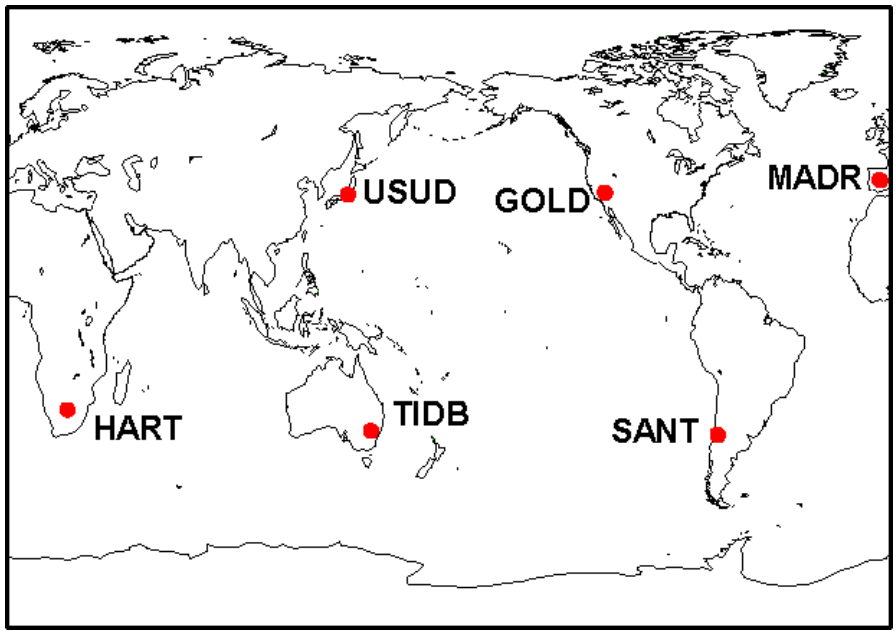


Figure C.2: Geographic location of the six stations that comprise the GPS ground network used for the simulations.

GPS Data

All simulations in this study made use of a 24 satellite GPS constellation with a ground network of 6 stations. It should be noted that more stations were available for processing, but experiments have shown that increasing the number of stations does not impact the results. Figure C.2 shows the geographic location of each GPS ground station used. The GPS satellite initial conditions and ground station locations were the same as those used by Sharma [52] in his studies. The GPS observations were sampled at 60 second intervals. In addition, the single phase-derived GPS range measurements had a Gaussian white noise applied with a 5mm standard deviation, translating into a 1 cm double-differenced noise level.

KBR Data

The inter-satellite K-band range (KBR) data were generated at 10 second intervals in order to provide adequate spatial distribution of the measurement points. The Nyquist sampling theory states that a signal needs to be sampled at twice its frequency in order for it to be fully recovered. This implies that a gravity of maximum degree $5600/10/2 = 280$ can be estimated with the given sampling interval. In practice, it is expected that a maximum degree of 140 to 160 is the most that can be supported by the GRACE data, so this sampling is more than adequate.

Three types of measurement errors were introduced to the inter-satellite range measurements. The three types of measurement errors included a system, oscillator and multipath noise. The system noise was introduced to represent errors in the receiver subsystem and was modeled as white noise. The oscillator noise was designed to replicate the drift errors inherent in the GRACE satellite's Ultra Stable Oscillator (USO). The oscillator noise was chosen to have a colored spectrum instead of white noise. This colored noise is a more accurate depiction of the true oscillator error, whose behavior is both spatially and frequency dependent. Lastly, the multipath error was added to account for the interference created by the indirect reflection of signals about the K-band antenna. Multipath errors are related to the angle that is created when the K-band antenna boresight and the line of sight (LOS) between the phase center of the two GRACE satellites are not parallel. An attitude variation time series was adopted to create a pessimistic multipath noise time series. Figure C.3 shows the power spectral density (PSD) of each of the measurement noise types, expressed in terms of range-rate, for a sample one day time span. It

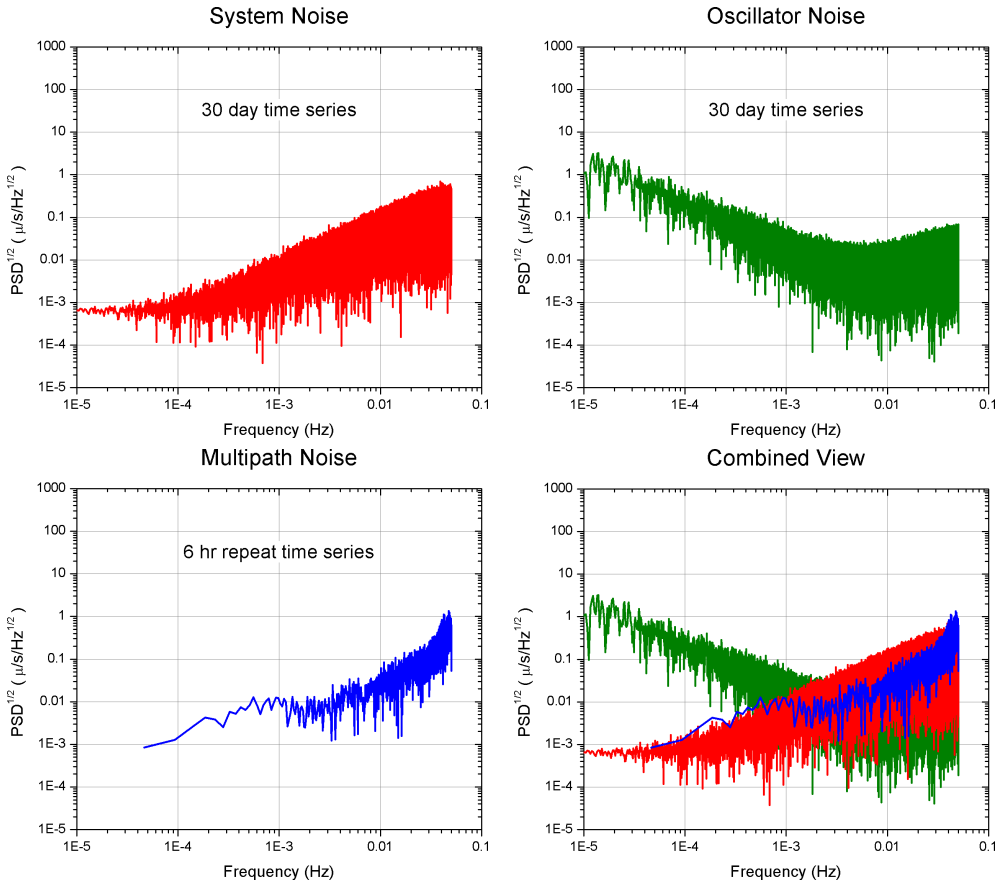


Figure C.3: Power spectral density of the simulated measurement noise inputs expressed in terms of range-rate.

should be noted that the attitude times series used to create the multipath errors in this work was different than that used by Kim, resulting in a different multipath error realization at the higher frequencies. In addition, the oscillator and system noise time series were implemented using a 30 day time series, while the multipath errors were created over a 6 hour interval which was then repeated. As figure C.3 illustrates, the oscillator noise is dominant at the low degrees, while the multipath and system errors are most powerful at the mid to high degrees. The time series of the range noise due to the oscillator drift is shown in Figure C.4, and illustrates the non-Gaussian behavior of the error

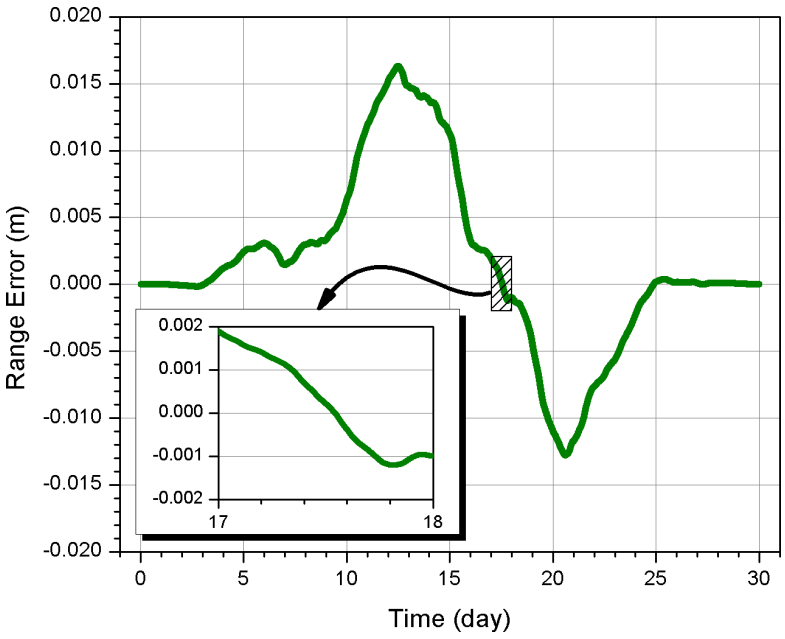


Figure C.4: Time series of the range error due to oscillator noise used in all of the simulations. The plot shows the variation over the 30 day time span as well as a sample one day interval (inset).

during the course of the 30 day simulation.

Non-gravitational Errors

When the simulated measurement observations are created, a standard set of errors are applied with regard to the non-gravitational forces acting on the satellite. These include atmospheric drag, solar radiation pressure and Earth radiation pressure. Later in the simulation creation process when the measurement partials are created, an additional random noise of $(1 + 0.005/f) \times 10^{-20}$ $m^2/s^4/Hz$ is applied to the accelerometer measurements so that these non-gravitational forces are not perfectly recovered.

Appendix D

Gravity Solution Tests

The evaluation a new gravity field solution is somewhat subjective, as there does not exist a definitive test that can be used to determine whether one particular field is better than another. Instead, we use a range of tests that, collectively, usually provide enough information to draw some general conclusions about the quality of a field. Below is a description of the various techniques used in this study to evaluate the various gravity fields generated from GRACE RL01 data.

D.1 Square Root Degree Variance

One of the simplest tests available to examine the quality of a field is to plot the square root of the degree variance, degree error variance, and degree difference variance. Given a set of normalized geopotential coefficents $(\overline{C}_{nm}, \overline{S}_{nm})$ and their respective standard deviations $(\delta\overline{C}_{nm}, \delta\overline{S}_{nm})$, the degree variance (DV) is calcuated as follows

$$DV_n = \sqrt{\sum_{m=0}^{n}(\overline{C}_{nm}^2 + \overline{S}_{nm}^2} \tag{D.1}$$

where n and m represent the spherical harmonic degree and order. Similarly, the formal errors, or degree error variance (DEV), is computed as

$$DEV_n = \sqrt{\sum_{m=0}^{n}(\delta\overline{C}_{nm}^2 + \delta\overline{S}_{nm}^2} \tag{D.2}$$

The degree difference variance (DDV) is often used to compare the estimates of two fields, and is generated by

$$DDV_n = \sqrt{\sum_{m=0}^{n} (\Delta\overline{C}_{nm}^2 + \Delta\overline{S}_{nm}^2} \tag{D.3}$$

where

$$\Delta\overline{C}_{nm} = (\overline{C}_{nm})_{field1} - (\overline{C}_{nm})_{field2}$$
$$\Delta\overline{S}_{nm} = (\overline{S}_{nm})_{field1} - (\overline{S}_{nm})_{field2}$$

The comparison field used for most DDV comparisons in this study was EGM96 [40], but TEG4 [59] can be substituted with similar results. The DV, DEV and DDV values are all scaled by the Earth's radius (6378136.3 km) in order to express the results in terms of geoid height.

D.2 Orbit Fit Test

Another test that is often a good indication of how a field performs at the low degrees is the orbit fit test. A gravity field is combined with existing SLR tracking data to estimate an orbit for a variety of different satellites, and the RMS from each resulting orbit fit is then computed. The better the gravity model, the better the fits. The use of additional drag and empirical once-per-revolution parameters can be used in the alongtrack and crosstrack directions (alongtrack only for LAGEOS 1/2) to improve the orbit fits, and can sometimes provide insight into the sensitivities of the gravity field being tested. The orbits are estimated across arcs ranging from 3-6 days, so they are most sensitive to the longer wavelength gravity signals. Below is a brief summary of the orbit

characteristics for the satellites used in the orbit tests (Source: the International Laser Ranging Service (ILRS), http://ilrs.gsfc.nasa.gov/).

Satellite	Inclination (deg)	Perigee (km)	Orbit Type
GEOS-3	115	824	Circular
GFZ-1	51.6	398	Circular
Lageos	101	5900	Circular
Lageos II	53	5600	Circular
Starlette	50	812	Circular
Stella	98	800	Circular, nearly sun-synch
Westpac	98	835	Circular, sun-synch

D.3 Ocean Circulation Tests

The ocean circulation test [62] is an excellent method of testing the performance of a given field in the mid degrees (i.e., $<$ degree 70). It does so by computing a (smoothed) dynamic ocean topography (DOT) map from an input gravity field (converted to a geoid) and existing satellite altimetry data. This DOT map is then differenced against available in situ data (i.e., ocean buoy measurements) for both zonal and meridional currents. A low RMS and high correlation indicate that the gravity field has produced an accurate geoid.

Bibliography

[1] Alpatov, P., Baker, G., Edwards, C., Gunnels, J., Morrow, G., Overfelt, J., van de Geijn, R., and Wu, Y.-J., "PLAPACK: Parallel Linear Algebra Package," in *Proceedings of the SIAM Parallel Processing Conference*, 1997.

[2] Anderson, E., Bai, Z., Demmel, J., Dongarra, J., DuCroz, J., Greenbaum, A., Hammarling, S., McKenney, A., Ostrouchov, S., and Sorensen, D., *LAPACK Users' Guide*, SIAM, Philadelphia, 1992.

[3] Bettadpur, S., *A Simulation Study of High Degree and Order Geopotential Determination using Satellite Gravity Gradiometry*, Ph.D. thesis, The University of Texas at Austin, Department of Aerospace Engineering and Engineering Mechanics, May 1993.

[4] Bischof, C. and van Loan, C., "The WY Representation for Products of Householder Matrices," *SIAM J. Sci. Stat. Comput.*, 8(1):s2–s13, Jan. 1987.

[5] Bjorck, A., *Numerical Methods for Least Squares Problems*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1996.

[6] Cheng, M., Gunter, B., Ries, J., Chambers, D., and Tapley, B., "Temporal Variation in the Earth's Gravity Field From SLR and CHAMP GPS Data," in *Proc. Of the 3rd Meeting of the International Gravity and Geoid Commission*, Thessaloniki, Greece, Aug. 2002.

[7] Choi, J., Dongarra, J., Pozo, R., and Walker, D., "ScaLAPACK: A Scalable Linear Algebra Library for Distributed Memory Concurrent Computers," in *Proceedings of the Fourth Symposium on the Frontiers of Massively Parallel Computation*, pp. 120–127, IEEE Comput. Soc. Press, 1992.

[8] Chtchelkanova, A., Gunnels, J., Morrow, G., Overfelt, J., and van de Geijn, R., "Parallel Implementation of BLAS: General Techniques for Level 3 BLAS," *Concurrency: Practice and Experience*, 9(9):837–857, Sept. 1997.

[9] Coleman, R., Leback, B., Norin, R., Scott, D., and van de Houten, K., "SOZ - A Dense, Out-of-Core Solver with Partial Pivoting for the iPSC/860: A Case History," in *1992 Annual Users Conference*, 1992.

[10] Condi, F., Gunter, B., Ries, J., and Tapley, B., "Combining Sea Surface and Terrestrial Gravity Data for Global Geopotential Modelling and Geoid Determination," *Eos Trans. AGU Fall Meet. Suppl.*, 84(46), 2003.

[11] Davis, G., *GPS-Based Precision Orbit Determination for Low Altitude Geodetic Satellites*, Ph.D. thesis, The University of Texas at Austin, Department of Aerospace Engineering and Engineering Mechanics, May 1996.

[12] D'Azevedo, E. and Dongarra, J., "The Design and Implementation of the Parallel Out-of-core ScaLAPACK LU, QR, and Cholesky Factorization Routines," LAPACK Working Note 118 CS-97-247, University of Tennessee, Knoxville, Jan. 1997.

[13] Dongarra, J., Du Croz, J., Hammarling, S., and Duff, I., "A Set of Level 3 Basic Linear Algebra Subprograms," *ACM Trans. Math. Soft.*, 16(1):1–17, March 1990.

[14] Dongarra, J., Du Croz, J., Hammarling, S., and Hanson, R., "An Extended Set of FORTRAN Basic Linear Algebra Subprograms," *ACM Trans. Math. Soft.*, 14(1):1–17, March 1988.

[15] Dongarra, J., Duff, I., Sorensen, D., and van der Vorst, H. A., *Solving Linear Systems on Vector and Shared Memory Computers*, SIAM, Philadelphia, PA, 1991.

[16] Dongarra, J., Moler, C., Bunch, J., and Stewart, G., *Linpack User's Guide*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1979.

[17] Dongarra, J., van de Geijn, R., and Walker, D., "Scalability Issues Affecting the Design of a Dense Linear Algebra Library," *J. Parallel Distrib. Comput.*, 22(3), Sept. 1994.

[18] Elmroth, E. and Gustavson, F., "New Serial and Parallel Recursive QR Factorization Algorithms for SMP Systems," in *PARA*, pp. 120–128, 1998.

[19] Elmroth, E. and Gustavson, F., "Applying Recursion to Serial and Parallel QR Factorization Leads to Better Performance," *IBM Journal of Research and Development*, 44(4):605–624, July 2000.

[20] European Space Agency (ESA), "The Solid-Earth Mission ARISTOTELES," ESA SP-329, International Workshop, Anacapri, Italy, Sept. 1991.

[21] European Space Agency (ESA), "Gravity Field and Steady-State Ocean Circulation Mission," *Report for Assessment of the Nine Candidate Earth Explorer Missions*, 1996.

[22] Golub, G. and van Loan, C., *Matrix Computations, 3rd Ed.*, The Johns Hopkins University Press, Baltimore, MD, 1996.

[23] Gropp, W., Lusk, E., and Skjellum, A., *Using MPI*, The MIT Press, 1994.

[24] Gunnels, J. A., Gustavson, F. G., Henry, G. M., and van de Geijn, R. A., "FLAME: Formal Linear Algebra Methods Environment," *ACM Trans. Math. Soft.*, 27(4):422–455, December 2001.

[25] Gunter, B., *Parallel Least Squares Analysis of Simulated GRACE Data*, Master's thesis, Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, 2000.

[26] Gunter, B., *AESoP User's Manual*, Center for Space Research, The University of Texas at Austin, 2004.

[27] Gunter, B., Quintana, E., and van de Geijn, R., "Parallel Out-of-Core LU and QR Factorization," The Eleventh SIAM Conference on Parallel Processing for Scientific Computing (PP04), Society for Industrial and Applied Mathematics (SIAM), San Francisco, CA, Feb. 2004.

[28] Gunter, B., Reiley, W., and van de Geijn, R., "Parallel Out-of-Core Cholesky and QR Factorizations with POOCLAPACK," in *Proceedings of the 15th International Parallel and Distributed Processing Symposium (IPDPS)*, IEEE Computer Society, 2001.

[29] Gunter, B., Tapley, B., and van de Geijn, R., "Advanced Parallel Least Squares Algorithms for GRACE Data Processing," in *Proc. of the International Association of Geodesy (IAG) Conference*, Budapest, Hungary, Sept. 2001.

[30] Gunter, B. and van de Geijn, R., "Parallel Out-of-Core Computation and Updating of the QR Factorization," *ACM Trans. Math. Soft.*, 31(1), March 2005 (To Appear).

[31] Gunter, B., van de Geijn, R., and Reiley, W., "Parallel Out-of-core Cholesky and QR Factorizations with POOCLAPACK," in *Proc. of the 15th International Parallel and Distributed Processing Symposium*, San Francisco, CA, April 2001.

[32] Hendrickson, B. and Womble, D., "The Torus-Wrap Mapping for Dense Matrix Calculations on Massively Parallel Computers," *SIAM J. Sci. Stat. Comput.*, 15(5):1201–1226, 1994.

[33] Householder, A., *The Theory of Matrices in Numerical Analysis*, Dover, New York, 1964.

[34] Jekeli, C., "Spherical Harmonic Analysis, Aliasing, and Filtering," *Journal of Geodesy*, 70:214–223, 1996.

[35] Kaula, W., *Theory of Satellite Geodesy*, Blaisdell Publishing Co., Waltham, MA, 1966.

[36] Keating, T., Taylor, P., Kahn, W., and Lerch, F., "Geopotential Research Mission, Science, Engineering and Program Summary," *NASA Technical Memorandum 86240*, 1986.

[37] Kim, J., *Simulation Study of a Low-Low Satellite-to-Satellite Tracking Mission*, Ph.D. thesis, Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, 2000.

[38] Klimkowski, K. and van de Geijn, R., "Anatomy of an out-of-core dense linear solver," in *Proceedings of the International Conference on Parallel Processing 1995*, vol. III - Algorithms and Applications, pp. 29–33, 1995.

[39] Lawson, C., Hanson, R., Kincaid, D., and Krogh, F., "Basic Linear Algebra Subprograms for Fortran Usage," *ACM Trans. Math. Soft.*, 5(3):308–323, Sept. 1979.

[40] Lemoine, F., Pavlis, E., Klosko, S., Pavlis, N., Chan, J., Kenyon, S., Trimmer, R., Salman, R., Rapp, R., and Nerem, R., "Latest Results from the Joint NASA GSFC and DMA Gravity Model Project," in *Suppl. to Eos Trans. AGU*, vol. 77, p. S41, 1996.

[41] Lichtenstein, W. and Johnsson, S., "Block-Cyclic Dense Linear Algebra," Tech. Rep. TR-04-92, Harvard University, Center for Research in Computing Technology, Jan. 1992.

[42] Quintana, E. and van de Geijn, R., "Formal Derivation of Algorithms for the Triangular Sylvester Equation," *ACM Trans. Math. Soft.*, 29(2):218–243, July 2003.

[43] Reigber, C., Luhr, H., and Schwintzer, P., "The CHAMP Geopotential Mission," *Boll. Geof. Teor. Appl.*, 40:285–289, 1999.

[44] Reiley, W., "Efficient Parallel Out-of-Core Implementation of the Cholesky Factorization," Tech. Rep. CS-TR-99-33, Department of Computer Sciences, The University of Texas at Austin, Dec. 1999, undergraduate Honors Thesis.

[45] Reiley, W. and van de Geijn, R., "POOCLAPACK: Parallel Out-of-Core Linear Algebra Package," Tech. Rep. CS-TR-99-33, Department of Computer Sciences, The University of Texas at Austin, Nov. 1999.

[46] Rim, H., *TOPEX Orbit Determination using GPS Tracking System*, Ph.D. thesis, The University of Texas at Austin, Department of Aerospace Engineering and Engineering Mechanics, Dec. 1992.

[47] Sanso, F., "On the Aliasing Problem in the Spherical Harmonic Analysis," *Bulletin Géodésique*, 64:313–330, 1990.

[48] Sanso, R. and Rummel, R. (Eds.), *Theory of Satellite Geodesy and Gravity Field Determination*, vol. 25 of *Lecture Notes in Earth Sciences*, Springer-Verlag, Berlin, 1989.

[49] Schreiber, R. and Loan, C. V., "A Storage-Efficient WY Representation for Products of Householder Transformations," *SIAM J. Sci. Stat. Comput.*, 10(1):53–57, Jan. 1989.

[50] Scott, D. S., "Parallel I/O and solving out-of-core systems of linear equations," in *Proceedings of the 1993 DAGS/PC Symposium*, pp. 123–130, Dartmouth Institute for Advanced Graduate Studies, Hanover, NH, June 1993.

[51] Seeber, G., *Satellite Geodesy, 2nd Ed.*, Walter de Gruyter, Berlin, 2003.

[52] Sharma, J., *Precise Determination of the Geopotential with a Low-Low Satellite-to-Satellite Tracking Mission*, Ph.D. thesis, The University of Texas at Austin, Department of Aerospace Engineering and Engineering Mechanics, Dec. 1995.

[53] Sneeuw, N. and Ilk, K., "The Status of Spaceborne Gravity Field Mission Concepts: A Comparative Simulation Study," in Segawa, J., Fujimoto, H., and Okubo, S. (Eds.), *Gravity, Geoid and Marine Geodesy*, IAG Symposium 117, pp. 171–178, Springer-Verlag, 1997.

[54] Sneeuw, N., Rummel, R., and Muller, J., "The Earth's Gravity Field from the STEP Mission," *Classical and Quantum Gravity*, (13):A113–A117, 1996.

[55] Snir, M., Otto, S. W., Huss-Lederman, S., Walker, D. W., and Dongarra, J., *MPI: The Complete Reference*, The MIT Press, 1996.

[56] Stewart, G., "Communication and matrix computations on large message passing systems," *Parallel Computing*, 16:27–40, 1990.

[57] Strazdins, P., "Optimal Load Balancing Techniques for Block-Cyclic Decompositions for Matrix Factorization," Tech. Rep. TR-CS-98-10, Canberra 0200 ACT, Australia, 1998.

[58] Tapley, B., "Statistical Orbit Determination Theory," *Advances in Dynamical Astronomy*, pp. 396–425, 1973.

[59] Tapley, B., Bettadpur, S., Chambers, D., Cheng, M., Choi, K., Gunter, B., Kang, Z., Kim, J., Nagel, P., Ries, J., Rim, H., Roesset, P., and Roundhill,

I., "Gravity Field Determination from CHAMP Using GPS Tracking and Accelerometer Data: Initial Results," *Eos Trans. AGU Fall Meet. Suppl.*, 82(47), 2001.

[60] Tapley, B., Bettadpur, S., Watkins, M., and Reigber, C., "The Gravity Recovery and Climate Experiment: Mission Overview and Early Results," *Geophysical Research Letters*, 31(9), May 2004.

[61] Tapley, B. and Born, G., "Sequential Estimation of the State and the Observation-Error Covariance Matrix," *AIAA Journal*, 9(2):212–217, 1971.

[62] Tapley, B., Chambers, D., Bettadpur, S., and Ries, J., "Large Scale Ocean Circulation from the GRACE GGM01 Geoid," *Geophys. Res. Lett.*, 30(22):2163, 2003, doi:10.1029/2003GL018622.

[63] Tapley, B., Schutz, B., and Born, G., *Statistical Orbit Determination*, Elsevier Academic Press, 2004.

[64] Tapley, B., Yuan, D., and Shum, C., "Gravity Field Determination and Error Assessment Techniques," in *Proceedings of the of the 1988 Chapman Conference on Progress in the Determination of the Earth's Gravity Field*, Ft. Lauderdale, Florida, Sept. 1988.

[65] Toledo, S. and Gustavson, F., "The Design and Implementation of SOLAR, a Portable Library for Scalable Out-of-Core Linear Algebra Computation," in *Proceedings of IOPADS '96*, 1996.

[66] Toledo, S. and Rabani, E., "Very Large Electronic Structure Calculations Using an Out-of-Core Filter-Diagonalization Method," *Journal of Computational Physics*, 180:256–269, 2002.

[67] van de Geijn, R., *Using PLAPACK: Parallel Linear Algebra Package*, The MIT Press, Cambridge, MA, 1997.

[68] van de Geijn, R. and Reiley, W., "POOCLAPACK: Parallel Out-of-core Linear Algebra Package," Tech. Rep. CS-TR-99-33, 1999.

[69] Watkins, D., *Fundamentals of Matrix Computations*, J. Wiley and Sons, New York, 1991.

[70] Watkins, M., Personal Communication, GRACE Science Data Systems Meeting, Austin, TX, Aug. 2002.

[71] Yuan, D.-N., *LLISS User's Manual*, Center for Space Research, The University of Texas at Austin, 1991.

[72] Yuan, D.-N., *The Determination and Error Assessment of the Earth's Gravity Field Model*, Ph.D. thesis, The University of Texas at Austin, Department of Aerospace Engineering and Engineering Mechanics, May 1991.

Vita

Brian Christopher Gunter was born in Albuquerque, New Mexico, on December 25, 1970, the son of Gary Lee Gunter and Ruth Laureen Gunter and brother of Sean Cary Gunter. After graduating from Arvada High School, Arvada, Colorado, in 1989, he subsequently attended Rice University for his undergraduate work. Graduating in 1994 with a Mechanical Engineering degree from Rice, he spent the next three years working as a computer consultant for the Houston-based firm BSG. He began his graduate studies at the University of Texas at Austin in the fall of 1997 and completed his Master of Science in Aerospace Engineering in August of 2000. He currently lives in Austin with his wife Angela and daughter Meghan.

Permanent address: 3501 Ambleside Dr.
Austin, Texas 78759

This dissertation was typeset with LaTeX[‡] by the author.

[‡]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.